# Boosted Object Detection Based on Local Features

by

**Haoyu Ren**

M.Sc., Chinese Academy of Science, 2010
B.Sc., TsingHua University, 2007

Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences

# Approval

| | |
|---|---|
| **Name:** | **Haoyu Ren** |
| **Degree:** | **Doctor of Philosophy (Computer Science)** |
| **Title:** | ***Boosted Object Detection Based on Local Features*** |
| **Examining Committee:** | **Chair:** Richard Vaughan<br>Associate Professor |

**Ze-Nian Li**
Senior Supervisor
Professor

**Mark Drew**
Supervisor
Professor

**Greg Mori**
Internal Examiner
Professor
School of Computing Science
Simon Fraser University

**Hao Jiang**
External Examiner
Associate Professor
Computer Science Department
Boston College

**Date Defended:** 7 April 2016

# Abstract

Object detection is to find and localize objects of a specific class in images or videos. This task is the foundation of image and video understanding, thus it becomes one of the most popular topics in the area of computer vision and pattern recognition. Object detection is not only essential for the study of computer vision, pattern recognition and image processing, but also valuable in the applications of public safety, entertainment and business. In this research, we aim to solve this problem in two focused areas: the local feature design, and the boosting learning.

Our research on local features could be summarized into a hierarchical structure with 3 levels. The features in different levels capture different object characteristic information. In the lower level, we investigate how to design effective binary features, which perform quite well for the object categories with small intra-class variations. In the middle level, we consider integrating the gradient information and structural information together. This results in more discriminative gradient features. In the higher level, we discuss how to construct the co-occurrence features. Using such features, we may get a classifier with high accuracy for general object detection.

After the feature extraction, boosted classifiers are learned for the final decision. We work on two aspects to improve the effectiveness of boosting learning. Firstly, we improve the discriminative ability of the weak classifiers by the proposed basis mapping. We show that learning in the mapped space is more effective compared to learning in the original space. In addition, we explore the efficiency-accuracy trade-off problem in boosting learning. The Generalization and Efficiency Balance (GEB) framework, and the hierarchical weak classifier are designed for this target. As a result, the resulting boosted classifiers not only achieve high accuracy, but also have good generalization and efficiency.

The performance of the proposed local features and boosting algorithms are evaluated using the benchmark datasets of faces, pedestrians, and general objects. The experimental results show that our work achieves better accuracy compared to the methods using traditional features and machine learning algorithms.

**Keywords:** Object Detection; Local Features; Boosting; Discriminative Ability; Generalization; Efficiency; Weak Classifier; Hierarchical; Binary Feature; Co-occurrence Feature; Basis Mapping; Efficiency-accuracy Trade-off

# Acknowledgements

First and foremost, I am sincerely grateful to my senior supervisor, Dr. Ze-Nian Li for his continuous support throughout my PhD studies. His contagious enthusiasm for science, life and friendship will be an inspiration for me forever. I have no idea how to show my appreciation to his kindnesss

I would like to truly thank my supervisor, Dr. Mark Drew for all his invaluable guidance and advice in my PhD study. I would also like to thank Dr. Greg Mori, Dr. Ping Tan, and Dr Hao Jiang for their help and suggestions for my research.

I had a great time at Simon Fraser University and my heartfelt thanks go to my dear friends Lei Chen, Xiaochuan Chen, Zhiwei Deng, Dr. Hossein Hajimirsadeghi, Jiawei He, Hexiang Hu, Dr. Ruizhen Hu, Moustafa S. Ibrahim, Mehran Khodabandeh, Jianqiao Li, Jinling Li, Xiaoyu Liu, Minlong Lu, Rui Ma, Ali Madooei, Srikanth Muralidharan, Dr. Nataliya Shapovalova, Shuyang Sun, Dr. Arash Vahdat, Dr. Wang Yan, Jordan Yap, Mengyao Zhai, Mingjing Zhang, Yatao Zhong, Dr. Guang-Tong Zhou and other friends of mine at Simon Fraser University.

I would also like to thank Dr. Xilin Chen, Dr. Shiguang Shan, Dr. Wei Zheng, and Dr. Xiaopeng Hong who had a great influence on me during my master study. Last but not least, I would like to express my gratitude to my family for all their love and encouragement throughout my life.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Research background

Object detection is one of the basic functions of human vision. The related technology is widely used in many applications, such as image indexing, object recognition, and driver assistant systems. Object detection is defined as a process which finds the location and size of any specific objects in the image or video sequence. It could be considered as a binary classification problem with the labels “object” and “non-object”. From the view of theoretical research, it belongs to the domain of image analysis and understanding, which is the foundation of the vision analysis. From the view of technologies, it is related with many domains such as pattern recognition, image processing, computer vision, and machine learning.

## 1.2 Main challenges

The main challenges of object detection come from two aspects: large intra-class variation and low Signal-to-Noise Ratio (SNR). Large intra-class variation means that the shape and appearance of the objects in the same category is very different, as shown in Fig. 1.1(a), the color and viewpoints are totally different for those cars, so that it is very difficult to

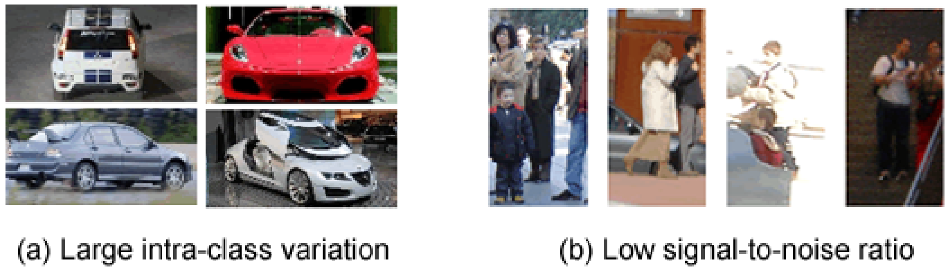

(a) Large intra-class variation
(b) Low signal-to-noise ratio

Figure 1.1: Challenges in object detection. (a) Large intra-class variation, (b) Low SNR.

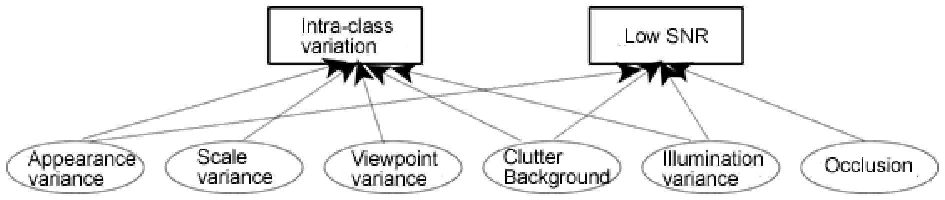


Figure 1.2: The relationship of the 6 difficulties and the two challenges in object detection.

describe what a car looks like. Low SNR means that compared to the background noise, the useful object information is very limited in the input image. As illustrated in Fig. 1.1(b), although the pedestrians have already been aligned based on the size and center location, it is still very difficult to distinguish the pedestrian due to the cluttered background. More specifically, these two challenges are related with 6 difficulties, including the appearance variance, the scale variance, the viewpoint variance, the cluttered background, the illumination variance, and the occlusion. The relationship of these difficulties and the two challenges are summarized in Fig. 1.2.

- The appearance variance is the main reason of the intra-class variation of the color and pose, as shown in Fig. 1.3(a). It is also one of the main reasons of low SNR.

- The scale variance mainly occurs in the creature categories, such as pedestrian, cow, cat, etc. In these categories, different objects have different part size and aspect ratio. As illustrated in Fig. 1.3(b), the ratio of head to body of a kid is 1/5, while it is 1/6 to 1/8 for a young woman.

- The viewpoint variance is another reason of the intra-class variation. It is one of the most difficult problems in object detection. In Fig. 1-3(c), the frontal view and side view of a motorbike are totally different. So we need to use different classifiers to deal with these two cases.

- The cluttered background will increase the difficulty of classifying the object with the background. As shown in Fig. 1.3(d), it is difficult to detect the bicycles due to the similar color with the surrounding background.

- The illumination variance is another reason of low SNR. When the illumination is weak, it will be relatively difficult to describe the details of the objects, as illustrated in Fig. 1.3(e).

- The occlusion is common in real object detection. It not only changes the appearance of the target objects, but also makes the alignment much harder. Some examples are shown in Fig. 1.3(f).

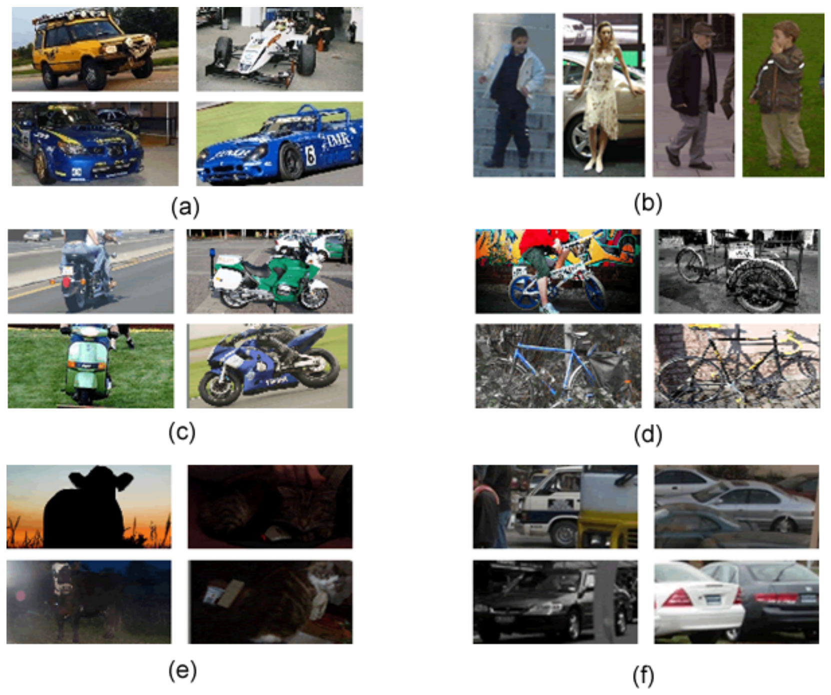


Figure 1.3: The difficulties in object detection. (a) appearance variance, (b) scale variance, (c) viewpoint variance, (d) cluttered background, (e) illumination variance, (f) occlusion.

## 1.3 Thesis contribution

### 1.3.1 Overview of my research

As mentioned in Chapter 1.2, the major challenges in object detection are the large intra-class variations and low SNR. To solve the intra-class variations problem, we focus on designing more discriminative and robust local features. More specifically, our research on local features could be summarized into three aspects, corresponding to different levels in the "feature hierarchy". It includes designing binary features, integrating the gradient information with structural information, and investigate the using of higher-order features, as shown in Fig. 1.4.

To solve the low SNR problem, we work on the boosted classifiers. Boosted classifiers are widely used due to their high efficiency and flexible structure. The discriminative ability of a boosted classifier is mainly decided by the weak classifier. So a feasible way is to strengthen the learning of the weak classifier. In addition, the efficiency is also an important issue in real object detection system. We plan to balance the efficiency-accuracy trade-off in boosting learning. Thus we expect to get the boosted classifiers with both high accuracy and good efficiency.

The overall framework of our boosted object detection is illustrated in Fig. 1.5. In the training procedure, a set of images are collected and each object is marked by a bounding box and the class label. Different local features are evaluated on current training samples

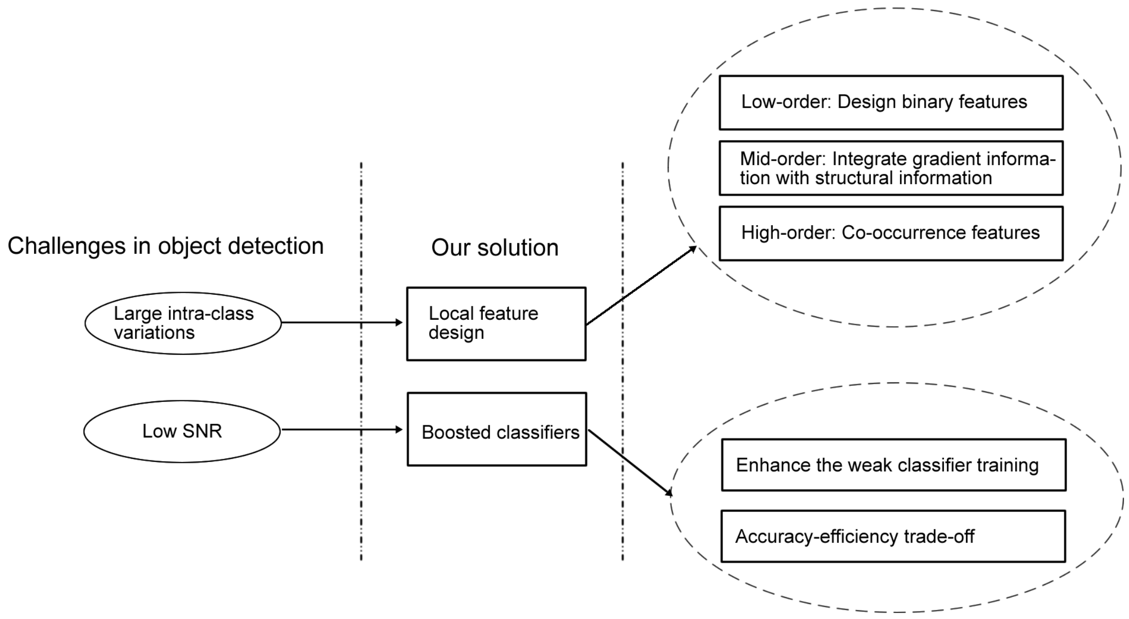

Challenges in object detection
Our solution
Large intra-class variations
Low SNR
Local feature design
Boosted classifiers
Low-order: Design binary features
Mid-order: Integrate gradient information with structural information
High-order: Co-occurrence features
Enhance the weak classifier training
Accuracy-efficiency trade-off

Figure 1.4: Overview of our approach.

to select the best one as the weak classifier. The final detector is constructed by concatenating all weak classifiers. In the detection procedure, given an input image, the first step is to apply the pre-processing modules such as histogram equalization or mean variance normalization. Then the scale pyramid is built, and the sliding window search strategy is adopted to go through the whole pyramid to evaluate all candidate windows. After the initial detection, a clustering algorithm will be applied to merge the initial results to the final output.

### 1.3.2 Our contributions on local features

1. For low-order features, we propose to design more discriminative binary features. Two works, the Boosted Local Binaries (BLB) [81] and Boosted Binary Patterns (BBP) [83] are published in ICME 2014 and ICME 2015 respectively. BLB and BBP represent the target object by a series of binary patterns, where each pattern consists of several rectangle pairs in different size and location. Different from traditional binary features which are constructed by fixed binary patterns, the scale and position of the rectangles in BLB and BBP are more flexible. In addition, these binary patterns are efficiently extracted in both the intensity domain and the gradient domain. Experimental results on public datasets demonstrate that these two features work well for object detection and tracking tasks. They are more discriminative compared to traditional binary features.

2. For middle-order features, we integrate the gradient information and local structural information together to improve the discriminative ability. Firstly, the Edged Histogram of Oriented Gradient feature (Edge-HOG) is published in ICIP 2014 [82]. Edge-HOG utilizes

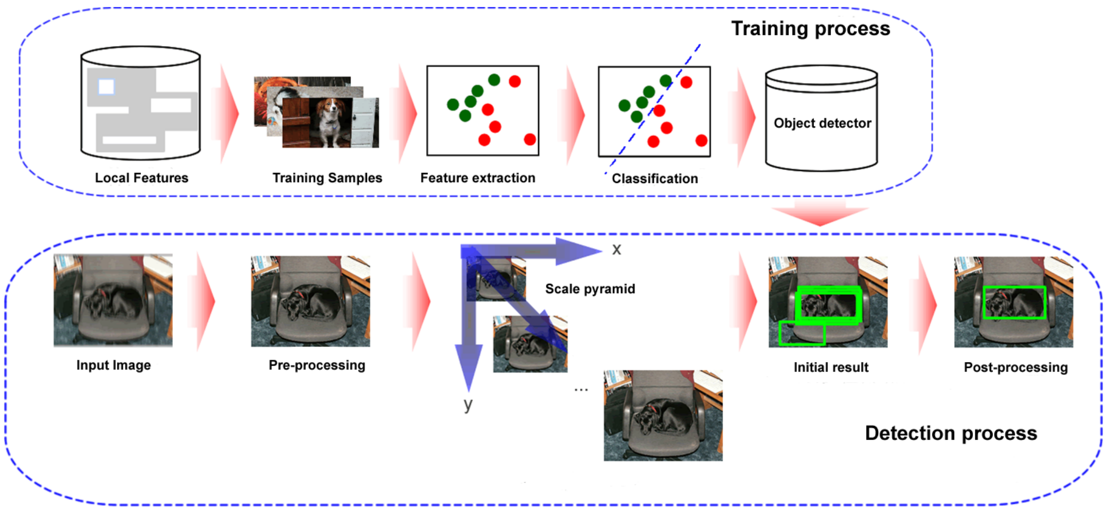


Figure 1.5: Overall framework of boosted object detection.

the histogram of oriented gradient of a series of regions as the base object description. These regions are concatenated along an edge, so that the local structure could be implicitly represented. We further design a method for extracting the structural information of a local region, which is based on the distance between the gravity centers and the geometric centers. The experimental results show that the Edge-HOG is more discriminative than the traditional gradient features. Secondly, we design a novel contour feature called Deformable Edge Set (DES) [84]. The DES consists of several Deformable Edge Features (DEF), which deform from an edge template to the actual object contour according to the distribution model of pixel gradients. The DES is constructed based on the combination of DEF, where the arrangement and the deformable parameters are learned in a subspace based on the local structural information. Experimental results show that the proposed DES not only locates the object bounding boxes, but captures the object contours as well.

3. For high-order features, we investigate the usage of the co-occurrence features in object detection. The co-occurrence features are defined as the distribution of co-occurring low-order information (e.g., gradient, intensity) of pixel pairs at a given offset. The 2D co-occurrence histogram will be utilized to describe the distribution of object characteristics. Using different low-order information, the co-occurrence features are able to capture the key characteristics of different object categories. In our work published in ICCV 2015 [85], we design three kinds of local co-occurrence features constructed by the traditional Haar, LBP, and HOG respectively. Then the boosted classifiers are learned, where each weak classifier corresponds to a local image region with a co-occurrence feature. Experimental results show that the performance of the resulting detector is competitive with the commonly-used methods for pedestrian detection and general object detection.

### 1.3.3 Our contributions on boosting algorithms

Our contributions on boosting could be summarized into two aspects:

1. To enhance the weak classifier learning, we propose a novel mapping method called basis mapping. This work is published in CVPR 2015 [80]. The key step is a non-linear mapping on original samples by referring to the basis samples before learning the weak classifiers, where the basis samples correspond to the hard samples in the current training stage. We show that the basis mapping based weak classifier is an approximation of kernel weak classifier while keeping the same computation cost as linear weak classifier. As a result, boosting with such weak classifiers is more effective. In our work, three different non-linear mappings are shown to work well. Experimental results show that the basis mapping consistently improves the detection accuracy and training efficiency of the boosted classifier. It achieves high performance on public datasets for both pedestrian detection and general object detection tasks.

2. The efficiency-accuracy trade-off is a classic topic in object detection. We propose two methods to solve this problem, the Generalization and Efficiency Balanced (GEB) framework, and the hierarchical weak classifier. The GEB framework is part of our published paper in ICCV 2015 [85]. In the feature selection procedure, the discriminative ability, the generalization power, and the computation cost of the candidate features are all evaluated for decision. Thus, boosting algorithm will arrange the efficient and robust features in the beginning stages, and highly discriminative features in the following stages. We know that the overall robustness and efficiency of a boosted detector is mainly decided by the beginning stages, which filter most of the negative samples. So the boosted classifiers trained by GEB could achieve both high accuracy and good efficiency. In our latest work of the hierarchical weak classifier, the boosted classifiers are constructed based on several kinds of local features. Each weak classifier corresponds to a local image region, from which several different types of features are organized in a hierarchical way. The weak classifier makes predictions by checking the features level by level. The decision will move to the next level when the current level is not confident enough. As a result, the boosted classifier using hierarchical weak classifiers has good efficiency even with various complicate features.

## 1.4 Thesis organization

After this introductory chapter, we provide an overview of the current research of object detection in Chapter 2. Chapter 3 introduces our work on low-order features, which includes the Boosted Binary Patterns and Boosted Local Binaries. In Chapter 4, we present two approaches to integrate the gradient information and the structural information. The Edge Histogram of Oriented Gradient and Deformable Edge Set will be briefly introduced. Chapter 5 describes the generalization form of the co-occurrence features, and several examples of how to extract co-occurrence vectors based on low-level features. Chapter 6 is the basis mapping, which improves the weak classifier learning procedure in boosting training. Chapter 7 consists of two work on the efficiency-accuracy trade-off: the Generalization-Efficiency Balance framework, and the hierarchical weak classifier. Finally, we conclude this research and propose the future work in Chapter 9.

# Chapter 2

# Recent progress on object detection

The related research on object detection could be divided into two parts: the feature extraction, and the classifier learning.

## 2.1 Research on feature extraction

The feature extraction is to generate the feature vectors from a given image to describe the object characteristic. These feature vectors could be extracted either from the whole image (global features) or from a specific region (local features). The feature vector reflects the capability of translating the object shape and appearance information into machine language. In general, the stronger discriminative ability the feature has, the better detection accuracy we will get. Designing effective features is also a feasible way to solve the scale variance, illumination variance, and cluttered background problem. In general, the features could be divided into 3 categories, binary features, gradient features, and high-order features.

### 2.1.1 Binary features

Binary features are constructed based on the binary coding of pixels or regions. Compared to other local features, binary features have lower discriminative power but better efficiency. They are widely used for the object categories whose intra-class variations are relatively small, such as frontal faces, or side-view cars. Haar features, which consist of two or more rectangular regions enclosed in a template, is one of the most successful binary features. Haar feature based face detector is proposed by Viola and Jones [114], which is the pioneer work of cascade object detection. Several Haar mutations are proposed in these years. Lienhart et al. [60] extended Haar features to different rotation angles. Mita et al. [66] jointed the basic Haar features to increase the discriminative ability. Park et al. [74] utilized the same prototype Haar feature but different extraction scheme to solve the illumination change in object detection.

Another popular binary feature Local Binary Pattern (LBP) is developed for texture classification [71] and the success is due to its computational simplicity and robustness under illumination variations. Yan et al. [130] replaced the single pixels in LBP with rectangles and utilized it in face detection. Ren et al. [81] further extended it to multi-locations and variable size. LBP is also combined with HOG feature because there two features are complimentary to each other. Wang et al. [117] firstly integrated HOG and LBP for pedestrian detection and achieved high detection rate on INRIA dataset. This work was extended in [135] and further improved in [118] with the combination of covariance matrix.

### 2.1.2 Gradient features

Gradient features extract the gradient magnitude and orientation to reflect the local edge characteristic. Compared to binary features, gradient features are more effective and robust. They work quite well for the object categories with clear contours. One of the typical gradient features is the edgelet feature [121]. It is mainly used to detect the pedestrians or cars with long edges and arcs in the contour. The extensions include the image strip feature [138], which extends single pixel to rectangles; and the Edge-HOG feature [82], which arranges the HOG rectangles in the same pattern as edgelet feature.

Scale Invariant Feature Transform (SIFT) [61] is invariant to image scaling, translation and rotation, and partially invariant to illumination changes and affine projection. Using SIFT feature, objects can be reliably recognized even from different views, low illumination or under occlusion. Another advantage is that some preprocessing stages such as the accurate alignment are not required using invariant features. The main disadvantage of SIFT is the low efficiency, which is about 3+ times slower compared to other commonly-used gradients features. To solve this problem, Bay et al. [8] proposed an accelerated version called SURF feature, relying on integral images and image convolutions. Another work is the ORB feature [88], which is also rotation invariant and partial resistant to occlusion.

Histogram of Oriented Gradient (HOG) [19] breaks the image region into a cell-block structure and generates histogram based on the gradient orientation and spatial location. Dalal & Triggs [19] proposed the basic form of the HOG feature with $2\times2$ cells. Multi-size versions were developed in [140][9][22], and further extended to pyramid structure [20][68][63][18][129]. HOG feature is also combined with other low-level features. Levi et al. [58] utilized an accelerated version of the feature synthesis method on the low-level description of multiple object parts. Bar-Hillel et al. [7] designed an iterative process including feature generation and pruning using multiple operators for part localization. Chen et al. [11] proposed the Multi-Order Contextual co-occurrence (MOCO), to implicitly model the high level context using solely detection responses from the object detection based on the combination of HOG and LBP. Paisitkriangkrai et al. [72] utilized the HOG built on the basis of low-level visual features combination and spatial pooling, which improved the translational invariance and thus the robustness of the detection process.

### 2.1.3 High-order features

More complicate features are summarized into the category of “High-order features”. The word “High-order” is in contrast to the low-order (intensity feature) and mid-order (gradient feature). One typical example is the covariance matrix [108], which is based on the covariance of low level features, e.g., the color vector, the norm of first and second derivatives of intensity, etc. Covariance matrices do not lie on Euclidean space, therefore the distance metric involving generalized eigenvalues which also follows from the Lie group structure of positive definite matrices is used. With this distance metric, covariance matrix works well on pedestrian detection [109] and general object detection [118].

Different from the covariance matrix, the co-occurrence features extract the characteristic of pixel pairs, such as the intensity contrast of two pixels, or the gradient orientation patterns of neighbouring pixels. According to whether the spatial neighbouring relationship among features is used in computing the co-occurrence statistics, existing co-occurrence features can be sorted into two categories: global co-occurrence features and local co-occurrence features. In [133], Yuan et al. proposed to mine co-occurrence statistics of SIFT words for visual recognition. Rasiwasia et al. [78] calculated the co-occurrence statistics on the whole image without considering the local spatial relationship among features. These works fall into the category of global co-occurrence features. In the following papers, the spatial co-occurrence are computed within locally adjacent neighbours instead of the whole image. Ito et al. [47] integrated the heterogeneous co-occurrence information of colour, edge and gradient information for object recognition. Yang et al. [131] proposed several pairwise co-occurrence features for food recognition. Xu et al. [127] designed the GMuLBP feature which detected co-occurrence orientation through gradient magnitude calculation. A rotation invariant version was proposed by Nosaka et al. [70] for texture classification and face recognition, and further improved by Qi. et al [76]. Chen et al. [11] proposed Multi-Order Contextual co-occurrence (MOCO), to implicitly model the high level context using solely detection responses.

## 2.2 Research on classifier learning

The classifier learning is to make the decision of whether an input image includes any objects or not. Training discriminative classifiers is a good way to solve the pose variance, viewpoint variance, and occlusion problem. There are two kinds of classifiers, the one based on the generative model, and the one based on the discriminative model. The discriminative model uses a map from the images to the class labels. Such classifiers could be denoted as $P(label|image, feature)$. The commonly-used discriminative classifiers include the Linear

Discriminative Analysis (LDA), the Support Vector Machine (SVM), and the Boosting. In contrast, the generative model uses a map from the class labels to the images. Such classifiers are in the format of $P(image|feature, label)P(label)$. Compared to discriminative model, the discriminative ability of the classifiers based on generative model is lower. The typical applications in object detection include the Naive Bayesian, the Restricted Boltzmann machine, and the mixture models.

### 2.2.1 Discriminative model based classifiers

LDA is used in statistics, pattern recognition and machine learning to find a linear combination of features which characterizes or separates two or more classes of objects or events. Given a set of feature vectors and its corresponding label set, LDA approaches the problem by assuming that the conditional probability density functions are both normally distributed with mean and covariance parameter respectively. In object detection, LDA is often applied together with histogram features. Laptev et al. [55] utilized LDA as weak learner for multi-valued histogram features and showed how to overcome problems of limited training. Xu et al. [128] adopted LDA on binary feature histograms in semi-supervised learning for object detection and classification. Qian et al. [77] integrated the features extracted from image content (i.e., color moment and edge direction histogram) and multi-classes LDA for social event classification.

The support vector machines are supervised learning models with associated learning algorithms that analyse data and recognize patterns, used for classification and regression analysis. Linear SVM training builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. Felzenszwalb et al. [28] described the Deformable Part Model (DPM) with Latent SVM for object detection, which is one of the best object detectors with traditional machine learning algorithms nowadays. Zhu et al. [139] presented a latent hierarchical structural learning method for object detection. An object was represented by a mixture of hierarchical tree models, where the model could be trained discriminatively using latent structural SVM learning. Appel et al. [21] utilized pyramid HOG feature and exploited the correlations of detector responses at nearby locations and scales by tightly coupling detector evaluation of nearby windows. Dollar et al. [20] computed finely sampled feature pyramids and utilized them in SVM weak classifier for pedestrian detection at a fraction of the cost, without sacrificing performance. Nam et al. [68] proposed an efficient feature transform that removed correlations in local neighborhoods. This resulted in an over-complete but locally decorrelated representation ideally suited for object detectors.

With the kernel trick, SVM is able to implicitly map the samples into the high dimensional space for classification. Kernel SVM replaces the dot product in linear SVM with the kernel functions, which allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. Hoang et al. [41] proposed a classification method based on a hybrid cascade boosting technique and RBF kernel SVM. Ren et al. [86] utilized the HIKSVM as weak classifier to detect pedestrians and cars. Due to the strong discriminative ability, kernel SVM achieves high accuracy. But the training and testing procedure is very slow. Maji [63] and Wu [124] investigate the efficient testing and training of additive kernel SVM respectively for object detection.

Boosting is a widely used machine learning algorithm for classification, which integrates several weak classifiers into a strong classifier. Y. Freund and R. Schapire proposed the AdaBoost (Adaptive Boosting) algorithm in 1995 [32]. They proved that the convergence of AdaBoost algorithm by adding constraint on the upper bound of the classification error. So AdaBoost algorithm will converge if the weak classifier is stronger than random guess. Inspired by this finding, several boosting algorithms corresponding to different weak classifiers and error functions are proposed. For example, the RealAdaBoost [93] is constructed using probability distribution function as weak classifier, the LogitBoost [33] utilizes log error function and logistic regression weak classifier. The modifications on the structure of strong classifier lead to other variants of boosting, such as the probabilistic boosting tree [107], the vector boosting [44], the cluster boosted tree [122], and the shrink boosting [40], etc.

### 2.2.2 Generative model based classifiers

In machine learning, naive Bayes classifiers apply Bayes' theorem with strong independence assumptions between the features. Since the classification ability of single naive Bayes classifier is relatively weak, the final decision is based on the combination of several naive Bayes classifiers. Grabner et al. [36] utilized online AdaBoost algorithm with naive Bayes classifier for real-time object tracking, where the feature vector composes of color, position, and gradient. Wu et al. [121] adopted the combination of the inferenced edge responses as feature vector and trained the pedestrian detection.

The restricted Boltzmann machine (RBM) is a generative stochastic artificial neural network that can learn a probability distribution over its set of inputs. It is a variant of Boltzmann machines, with the restriction that their neurons must form a bipartite graph. This restriction allows for more efficient training algorithms than are available for the general class of Boltzmann machines, in particular the gradient-based contrastive divergence algorithm. Eslami et al. [26] proposed a type of deep Boltzmann machine that called Shape Boltzmann Machine (SBM) for the task of modelling foreground/background and parts-based shape images. Tang et al. [105] introduced the Robust Boltzmann Machine (RoBM), which allowed Boltzmann Machines to be robust to corruptions. In the domain of visual recognition, the RoBM was able to accurately deal with occlusions and noise by using multiplicative gating to induce a scale mixture of Gaussian functions over pixels. Srivastava et al. [101] proposed a Deep Boltzmann Machine for learning a generative model of multi-model data. The model defined a probability density over the space of multi-model inputs.

By sampling from the conditional distributions over each data modality, it was possible to create the representation even when some data modalities are missing.

The mixture model is a probabilistic model for representing the presence of a subset within the whole object category. A mixture model corresponds to the mixture distribution that represents the probability distribution of the objects. The typical mixture models include Gaussian Mixture Model (GMM), multivariate Gaussian mixture model, and Categorical mixture model. Cheng et al. [14] introduced a regional contrast based object detection algorithm, which simultaneously evaluated global contrast differences and spatial weighted coherence scores based on GMM. Spinello et al. [99] proposed a novel adaptive fusion approach for object detection based on hierarchical mixtures models. The hierarchy was a two-tier architecture that for each modality, each frame and each detection computed a weight function using Multivariate GMM that reflected the confidence of the respective information.

### 2.2.3 Deep learning based object detection

In recent years, the neural network becomes one of the hottest topics in object detection. Due to the good description ability of deep learning features, the resulting classifier achieves high accuracy on general object detection. Among the deep neural networks, the Convolutional Neural Network (CNN) is most commonly-used. Sermanet et al. [95] used convolutional networks for PASCAL-style object detection concurrent with the development of R-CNNs. Szegedy et al. [102] modeled object detection as a regression problem. Given an image window, they used a CNN to predict foreground pixels over a coarse grid for the whole object as well as the object's top, bottom, left and right halves. Agrawal et al. [1] trained an R-CNN from a random initialization on VOC 2007 trainval. They achieved a mAP of 40.7% on VOC 2007 test with using only half the amount of training data as [102]. He et al. [39] improved R-CNN efficiency by sharing computation through a feature pyramid, allowing for detection at a few frames per second. Girshick [34] showed that it is possible to further reduce training and testing times, while improving detection accuracy and simplifying the training process, using an approach called "Fast R-CNN".

Some other researchers work on the detection process of the neural networks. We know that the dominant approach to object detection has been based on sliding-window detectors. An alternative is to first compute a pool of image regions, each serving as a candidate object, and then to filter these candidates in a way that aims to retain only the true objects. Hoiem et al. [42] used multiple segmentation hypotheses to estimate the rough geometric scene structure. Russell et al. [90] extended it to automatically discover object classes in a set of images. Van de Sande et al. [110] further proposed an improve version called the "selective search" for object detection by showing strong results on PASCAL object detection. Regarding the post-processing in the detection, EdgeBoxes [**?**] outputed high-quality rectangular (box) proposals quickly. BING [15] generated box proposals at 3 ms per image, but the quality is relatively poor. Other methods focus on pixel-wise segmentation, producing regions instead of boxes. These approaches include RIGOR [46] and MCG [5], which took 10 to 30ms per image and GOP [53], a faster methods that took  1s per image.

R-CNNs have been extended to a variety of new tasks and datasets. Karpathy et al. [50] learned a model for bi-directional image and sentence retrieval. Their image representation was derived from an R-CNN trained to detect 200 classes on the ILSVRC2013 detection dataset. Gkioxari et al. [35] used multi-task learning to train R-CNNs for person detection, 2D pose estimation, and action recognition. Hariharan et al. [38] proposed a unification of the object detection and semantic segmentation tasks, and trained a two-column R-CNN for this task. They showed that a single region proposal algorithm (MCG [5]) can be used effectively for traditional boundingbox detection as well as semantic segmentation. Gupta et al. [37] extended R-CNNs to object detection in depth images. They showed that a well designed input signal, where the depth map is augmented with height above ground and local surface orientation with respect to gravity, allowed training an R-CNN that outperformed existing RGB-D object detection baselines. Song et al. [98] trained an R-CNN using weak, image-level supervision by mining for positive training examples using a submodular cover algorithm and then training a latent SVM. Many systems based on, or implementing, R-CNNs were used in the recent ILSVRC2014 object detection challenge [89], resulting in substantial improvements in detection accuracy. In particular, the winning method, GoogleNet [104][103], used an innovative network design in an R-CNN. With a single network, they improved R-CNN performance to 38.0% mAP from a baseline of 34.5%. They also showed that an ensemble of six networks improves their result to 43.9% mAP.

## 2.3 AdaBoost Classifier Learning

Among the above classifier learning algorithms, AdaBoost is commonly-used for object detection. AdaBoost is a machine learning algorithm formulated by Yoav Freund and Robert Schapire. It can be used in conjunction with many other types of learning algorithms to improve their performance. The output of the other learning algorithms (weak classifiers) is combined into a weighted sum that represents the final output of the boosted classifier. In some problems, however, it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing (e.g., their error rate is smaller than 0.5 for binary classification), the final model can be proven to converge to a strong classifier.

### 2.3.1 Discrete AdaBoost

AdaBoost refers to a particular method of training a boosted classifier. Suppose we have a data set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each item $\mathbf{x}_i$ has an associated class $y_i \in \{-1, 1\}$, and a set of weak classifiers $\{h_1, \ldots, h_T\}$, each of which outputs a hypothesis $d = h_t(\mathbf{x}_i) \in \{-1, 1\}$ for each item. A boosted classifier is in the following form

$$H_t(\mathbf{x}) = \sum_{t=1}^{T} h_t(\mathbf{x}), \tag{2.1}$$

where each $h_t$ is a weak classifier that takes an object (the feature vector) $\mathbf{x}$ as input and returns a real valued result indicating the class of the object. The sign of the weak classifier output identifies the predicted object class and the absolute value gives the confidence in that classification. Similarly, the T-layer classifier will be positive if the sample is believed to be in the positive class and negative otherwise.

At each iteration $t$, a weak classifier is selected and assigned a coefficient $\alpha_t$ such that the sum training error $E_t$ of the resulting T-stage boosted classifier is minimized

$$E_t = \sum_{i} H_{t-1}(\mathbf{x}) + \alpha_t d_i. \tag{2.2}$$

Here $H_{t-1}(\mathbf{x})$ is the boosted classifier that has been built up to the previous stage of training, $E(H)$ is some error function and $h_t(\mathbf{x}) = \alpha_t d$ is the weak classifier that is being considered for addition to the final strong classifier. In each iteration of the training process, a weight is assigned to each sample in the training set equal to the current error $E(H_{t-1}(\mathbf{x}_i))$ on that sample. These weights can be used to inform the training of the weak classifier.

In AdaBoost, the error function in Eq. 2.2 is set as the sum of its exponential loss on each data point, given as follows

$$E = \sum_{i=1}^{T} e^{-y_i H_t(\mathbf{x}_i)}. \tag{2.3}$$

Suppose all the weight at first stage is 1, and the weight of sample $i$ at state $t$ is $w_i^t = e^{-y_i h_{t-1}(\mathbf{x}_i)}$, we have

$$E = \sum_{i=1}^{T} w_i^t e^{-y_i \alpha_t h_t(\mathbf{x}_i)}. \tag{2.4}$$

We can split this summation between those data points that are correctly classified and those which are misclassified as

$$E = \sum_{y_i = h_t(\mathbf{x}_i)} w_i^t e^{-\alpha_t} + \sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^t e^{\alpha_t}. \tag{2.5}$$

Since the only part of the right-hand side of this equation that depends on $h_t$ is $\sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^t$, we see that the $h_t$ that minimizes $E$ is just the one that minimizes $\sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^{(t)}$. In order to determine the desired weight $\alpha_t$ that minimizes $E$ with the $h_t$ that we just determined, we differentiate

```
Parameters
  N   number of training samples
  T   maximum number of weak classifiers
Input: Training set $\{(\mathbf{x}_i, y_i)\}, y_i \in \{-1, 1\}$
1. Initialize sample weight and classifier output
   $w_i = 1/N, H(\mathbf{x}_i) = 0$
2. Repeat for $t = 1, 2, \ldots, T$
   2.1 Train the weak classifier $h_t$ using current training samples
   2.2 Get the classification error $\epsilon_t = \sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^t / \sum_{i=1}^N w_i^t$
   2.3 Select the minimum error, and caculate $\alpha_t$
   2.4 Update sample weight $w_i^{t+1} = \frac{1}{Z} w_i^t e^{-\alpha_t y_i h_t(\mathbf{x}_i)}$, $Z$ is the normalization factor
   2.5 Update strong classifier $H_{t+1}(\mathbf{x}_i) = H_t(\mathbf{x}_i) + h_t(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^T h_j(\mathbf{x})]$
```

Figure 2.1: Discrete AdaBoost Training.

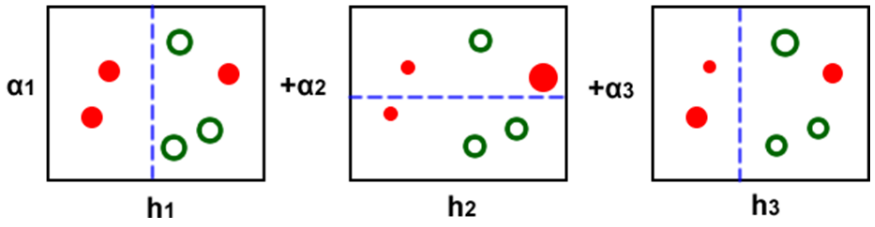


Figure 2.2: Sample weight update in AdaBoost.

$$\frac{dE}{d\alpha_t} = \sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^t e^{\alpha_t} - \sum_{y_i = h_t(\mathbf{x}_i)} w_i^t e^{-\alpha_t}. \tag{2.6}$$

Setting this to zero and solving for $\alpha_t$ yields

$$\alpha_t = \frac{1}{2} \ln \left( \frac{\sum_{y_i = h_t(\mathbf{x}_i)} w_i^t}{\sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^t} \right) \tag{2.7}$$

We calculate the weighted error rate of the weak classifier to be $\epsilon_t = \sum_{y_i \neq h_t(\mathbf{x}_i)} w_i^t / \sum_{i=1}^N w_i^t$, so it follows that

$$\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right). \tag{2.8}$$

which is the negative logit function multiplied by 0.5. The workflow of the whole discrete AdaBoost is illustrated in Fig. 2.1. AdaBoost will reduce the weight of mis-classified samples (difficult samples), and increase the weight of correct-classified samples (easy samples), as illustrated in Fig. 2.2. The final strong classifier is the linear combination of several weak classifiers.

```
Parameters
  N   number of training samples
  T   maximum number of weak classifiers
Input: Training set $\{(\mathbf{x}_i, y_i)\}, y_i \in \{-1, 1\}$
1. Initialize sample weight and classifier output
   $w_i = 1/N, H(\mathbf{x}_i) = 0$
2. Repeat for $t = 1, 2, \ldots, T$
   2.1 Update the sample weight $w_i$ using the $h^{th}$ weak classifier output $w_i = w_i e^{-y_i h_t(\mathbf{x}_i)}$
   2.2 Build the predict distribution function $W_+$ and $W_-$
   2.3 Select the best feature with the lowest classification error $Z$
   2.4 Update weak classifier $h_t(x)$
   2.5 Update strong classifier $H_{t+1}(\mathbf{x}_i) = H_t(\mathbf{x}_i) + h_t(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^{T} h_j(\mathbf{x})]$
```

Figure 2.3: RealAdaBoost Training.

### 2.3.2 RealAdaBoost

The output of discrete AdaBoost is either -1 or 1, so the classification ability is relatively limited. Schapire et al. [94] extend the output to real numbers, which results in the RealAdaBoost. In RealAdaBoost, each feature can be seen as a function from the image space to a real valued range. The sample space is divided into $N_b$ equal sized sub-ranges $B_1, \ldots, B_{N_b}$, and the weak classifier is defined as a piecewise function

$$h(\mathbf{x}) = \frac{1}{2} ln(\frac{W_+^j + \epsilon}{W_-^j + \epsilon}), \tag{2.9}$$

where $\epsilon$ is the smoothing factor, $W_\pm$ is the probability distribution of the feature response for positive/negative samples, implemented as a histogram

$$W_\pm^j = P(\mathbf{x} \in B_j, y \in \{-1, 1\}), j = 1, \ldots, N_b. \tag{2.10}$$

The best feature is selected according to the classification error Z of the piecewise function. Better features lead to lower $Z$

$$Z = 2 \sum_j \sqrt{W_+^j W_-^j}. \tag{2.11}$$

The workflow of the RealAdaBoost is illustrated in Fig. 2.3.

### 2.3.3 LogitBoost

LogitBoost [33] represents an application of established logistic regression techniques to the AdaBoost method. Suppose there is a binary classification problem, which classifies the

```
Parameters
  $N$    number of training samples
  $T$    maximum number of weak classifiers
Input: Training set $\{(\mathbf{x}_i, y_i)\}, y_i \in \{0,1\}$
1. Initialization    $w_i = 1/N, H(\mathbf{x}_i) = 0, p(\mathbf{x}_i) = 0.5$
2. Repeat for $t = 1, 2, \ldots, T$
  2.1 Compute $z_i$ and $w_i$
  2.2 Fit the function $h_t$ by weighted least square regression from $\mathbf{x}$ to $z_i$
  2.3 Update $H(\mathbf{x}_i)$ and $p(\mathbf{x}_i)$
  2.4 Update strong classifier $H_{t+1}(\mathbf{x}_i) = H_t(\mathbf{x}_i) + h_t(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^{T} h_j(\mathbf{x})]$
```

Figure 2.4: LogitBoost training.

sample $\mathbf{x}$ to class $\{0, 1\}$. In LogitBoost, the probability of sample $\mathbf{x}_i$ being a member of class 1 is represented by

$$p(\mathbf{x}_i) = \frac{e^{H(\mathbf{x}_i)}}{e^{H(\mathbf{x}_i)} + e^{-H(\mathbf{x}_i)}}, \tag{2.12}$$

where $H$ is the strong classifier

$$H(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{T} h_j(\mathbf{x}). \tag{2.13}$$

The LogitBoost algorithm treats the weak classifiers as a set of regression functions $h_j(\mathbf{x}_i)_{j=1,2,,T}$ by minimizing the negative binomial log-likelihood of the training samples $l(y, p(\mathbf{x}_i))$ through Newton iterations

$$l(y, p(\mathbf{x}_i)) = -\sum_{i=1}^{N} [y_i log(p(\mathbf{x}_i)) + (1 - y_i) log(1 - p(\mathbf{x}_i))]. \tag{2.14}$$

This regression function $h_j(\mathbf{x}_i)_{j=1,2,,T}$ fits the training samples $\mathbf{x}_i$ to response values $z_i$

$$z_i = \frac{y - p(\mathbf{x}_i)}{p(\mathbf{x}_i)(1 - p(\mathbf{x}_i))}. \tag{2.15}$$

After regression, $H(\mathbf{x}_i)$ and $p(\mathbf{x}_i)$ are updated according to Eq. 2.13 and Eq. 2.12. Then the sample weights are updated as

$$w_i = p(\mathbf{x}_i)(1 - p(\mathbf{x}_i)). \tag{2.16}$$

Fig. 2.4 illustrates the detailed algorithm.

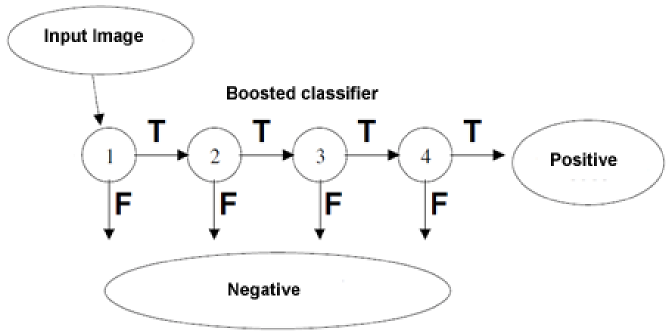


Figure 2.5: Cascade classifiers.

### 2.3.4 Cascade boosted classifier

Cascade classifier is a particular case of ensemble learning based on the concatenation of several classifiers, using all information collected from the output from a given classifier as additional information for the next classifier in the cascade. As illustrate in Fig. 2.5, the strong classifier consists of several weak classifiers. A candidate sample will be detected as positive sample if and only if it goes through all the weak classifiers. Since most of the negative samples will be rejected by the first several weak classifiers, the overall detection process will be quite efficient.

The training procedure of the cascade boosted classifier follows the state-by-state strategy. Each stage consists of several weak classifiers selected by the above boosting algorithms. The termination of the training of each stage is decided by the current false positive rate. For example, if the reject threshold of each stage is set to 0.5, the training will stop after it reject 50% of the negative samples.

Since some negative samples are rejected in each stage, we need to fill this gap before training the next stage. This process is called bootstrap. It is achieved by running the previous stages on a large negative set. Since the false positive rate will be reduced stage by stage, the bootstrap will take more and more time.

The cascade classifiers could be generalized to any machine learning algorithms. All the boosted classifiers in this thesis follow this structure.

## 2.4 Conclusion

In this chapter, we briefly introduced the commonly-used local features and machine learning methods for object detection. Traditional local features show considerable accuracy for some object categories. Unfortunately, these features have some drawbacks. The major problem for binary features is the fixed binary pattern. The size and position of the pixel and region pair are fixed, which will limite the discriminative ability. Using more flexible patterns for feature extraction leads to better object description. The gradient features extract the gradient statistic information as the feature vector, which is relatively robust to some noise. Unfortunately, these features ignore some important structural information in the histogram calculation procedure. Suppose there are two rectangles, each one contains only one edge. The length and orientation of these two edges are exactly the same, but the position inside the rectangle is different. In this case, most of the traditional gradient features will output the same feature vector. Integrating the structural information with the gradient vector is a feasible way to solve this problem. Regarding the high-order features, the existing methods propose to enumerate all possible patterns and extract a high-dimensional feature vector, which is rather time consuming for real-time applications. It might also bring useless information in this dense extraction procedure. Localizing these high-order features and filtering the useless dimensions will be helpful.

On the other hand, the boosted classifier is widely used in object detection. Traditional boosting algorithms such as AdaBoost or LogitBoost achieve high accuracy for some simple detection tasks, such as text and faces. For more complicate object categories, the detection accuracy is still not sufficient. Improving the effectiveness of boosting methods is always an active topic. In addition, most of the research on the discriminative model based classifier emphasize the accuracy only. Since the efficiency is one of the basic requirements of real-time applications, it will be helpful to deal with the accuracy-efficiency trade-off. Some existing methods try to solve this problem by prior knowledge [126][123], which is limited to the detection tasks. Solving this trade-off problem in general object detection by some adaptive methods is a valuable topic.

# Chapter 3

# Low-order features

We divide the local features into a “feature hierarchy”. The feature hierarchy consists of three levels, the low-order features, the mid-order features, and the high-order features, as shown in Fig. 3.1. From lower level to higher level, the discriminative ability is increased, but the efficiency is reduced.

Low-order features extract the feature vector based on intensity information. Most of the binary features belong to this category. Compared to other local features, low-order features have lower discriminative power but better efficiency. They are widely used for the object detection tasks whose intra-class variation is relatively small, such as frontal face detection, or side-view car detection. In this level, we focus on designing effective binary features. This results in two works, the Boosted Local Binary (BLB), and Boosted Binary Patterns (BBP).

## 3.1 Boosted Local Binaries

Due to the high efficiency, binary feature is one of the most commonly-used features in object detection. The existing binary features such as the Haar [114], LBP [71], or LAB [130] has been demonstrated on texture analysis, object detection, and face recognition. Despite its simplicity, a number of the binary feature modifications and extensions have been proposed. Some of them focus on the post-processing steps which improve the discrimination ability of binary coding [51]. But the computation cost will also be increased. Others focus on the definition of the location where gray value measurement is taken [69]. Such improvement on the discriminative ability is relatively limited because these locations are artificially designed, and using intensity information might not be sufficient to solve some complex object detection problems.

We introduce the Boosted Local Binary (BLB) feature in ICME 2014 [81], where the local region pairs are selected by RealAdaBoost algorithm considering both the discriminative ability and the feature structure diversity. In addition, we identify that using the

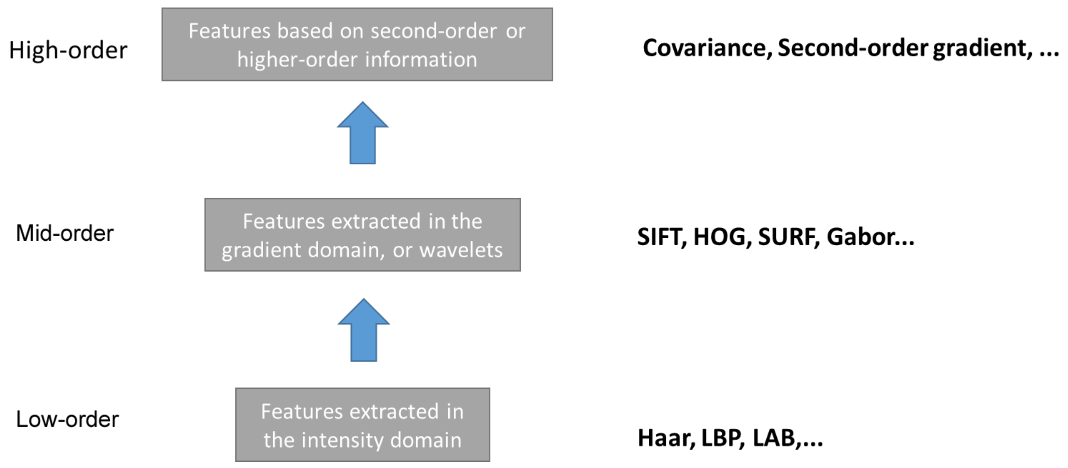

High-order
Features based on second-order or higher-order information
Covariance, Second-order gradient, ...
Mid-order
Features extracted in the gradient domain, or wavelets
SIFT, HOG, SURF, Gabor...
Low-order
Features extracted in the intensity domain
Haar, LBP, LAB,...

Figure 3.1: Local feature hierarchy.

gradient image and gray image together for binary coding is more effective than only using the gray image. As a result, the proposed BLB feature is more discriminative and robust compared to common-used binary features.

### 3.1.1 Traditional Binary Features

LBP [71] is developed for texture classification and the success is due to its computational simplicity and robustness under illumination variations. The traditional LBP is constructed by the binary coding of the intensity contrast of a center pixel and several surrounding pixels. If the intensity of the surrounding pixels is higher than the center one, the corresponding bit will be assigned 1, otherwise it will be assigned 0. Given a center pixel $c$, the number of surrounding pixels $d$, and the distance $r$ between the center pixel and the surrounding pixels $i_1, \ldots, i_d$, the LBP response is denoted by $LBP_{d,r}$, defined by Eq. 3.1

$$LBP_{d,r} = \sum_{j=1}^{d} sign(I_{i_j} - I_c) \times 2^{j-1}, \quad r = dis(c, i_j) \tag{3.1}$$

where $sign(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$, $I$ is the intensity, $dis$ is the Euclidean distance. Fig. 3.2 illustrates an example of LBP with $d = 8, r = 1$.

Local Assembled Binaries (LAB) [130] replaces single pixels with rectangles, as shown in Fig. 3.2. The encoding scheme is the same: if the intensity sum of the surrounding rectangle $R_1, \ldots, R_d$ is larger than the center rectangle $C$, the corresponding bit will be assigned 1; otherwise it will be assigned 0. In this equation, $I_C$ is the intensity sum of the pixels in $C$.

$$LAB = \sum_{i=1}^{d} sign(I_{R_i} - I_C) \times 2^{i-1}. \tag{3.2}$$

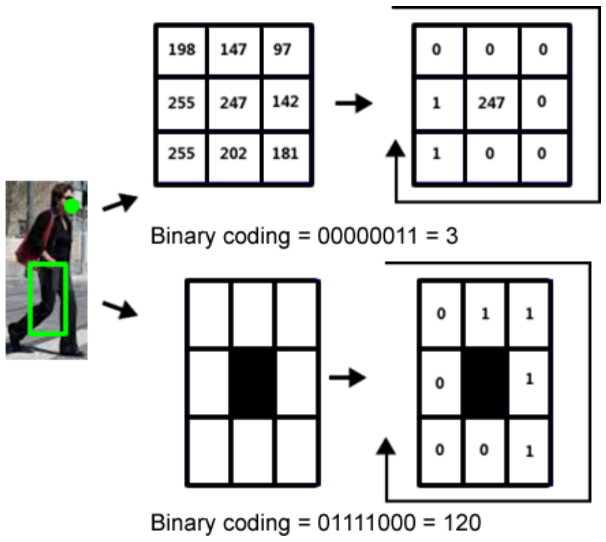


Figure 3.2: $LBP_{8,1}$ feature (top) and LAB feature (bottom).

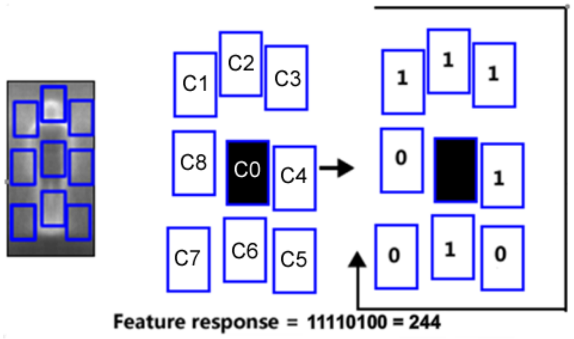


Figure 3.3: Boosted local binaries.

The computation of LBP is efficient because the binary coding could be calculated by bit-shift operations. Although LAB uses rectangles instead of single pixels, the intensity sum in any rectangles could also be easily extracted by the integral image [114].

### 3.1.2 Boosted Local Binaries (BLB)

The above traditional binary features have certain drawbacks when employed to encode general object's appearance. A notable disadvantage is the insufficient discriminative ability. The feature response of LBP and LAB depend on the intensities of particular locations and thus vary by object's appearance. It will be easily influenced by illumination, occlusion, and noises. In addition, although the size of the rectangles of LAB is flexible, the patterns of local pixel and adjacent rectangles are fixed. It might not have sufficient ability to describe the objects in some complicate detection tasks.

Boosted Local Binaries (BLB) comprises of 8 surrounding rectangles $C_1, C_2, \ldots, C_8$ and the center rectangle $C_0$ with the same size, as illustrated in Fig. 3.3. The centers of

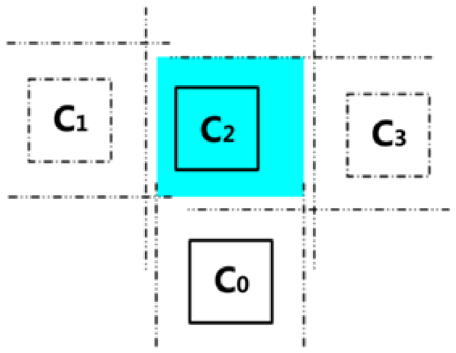


Figure 3.4: Surrounding rectangles in BLB.

$C_1, C_2, \ldots, C_8$ are arranged in a similar order to LBP and LAB, and none of them are overlapped. Fig. 3.4 shows the center rectangle $C_0$, left-top rectangle $C_1$, top rectangle $C_2$ and right-top rectangle $C_3$. $C_2$ is located at the left side of $C_3$, and $C_1$ is located at the left side of $C_2$. All of these three rectangles lay at the top of $C_0$. In addition, to keep the neighboring patterns, each neighboring pair (e.g., $C_1$ and $C_2$, $C_2$ and $C_0$, $C_2$ and $C_3$) should overlap at least 50% either on their width or on their height. In Fig. 3.4, the dot-dashed lines around each rectangle reflect this possible region of its neighbors according to this constraint. So the possible region for $C_2$ is the cyan region which is intersected by the corresponding dot-dashed lines of $C_0$, $C_1$ and $C_3$.

If the surrounding rectangles lay far away from the center rectangle, the feature response will be easily influenced by noises. It means that the classification confidence based on this feature might be lower. So we define the structure diversity $D$ of the BLB as Eq. 3.3

$$D = \frac{1}{8} \sum_{i=1}^{8} \frac{2dis(C_i, C_0)}{w_i + h_i}. \tag{3.3}$$

The $dis(C_i, C_0)$ denotes the Euclidean distance between the centers of rectangle $C_i$ and rectangle $C_0$. $w_i$ and $h_i$ are the width and height of the rectangle $C_i$. This factor will be used in the penalty term of the feature selection procedure.

Besides using these patterns on intensity image, we also apply them on gradient images. In consideration of the efficiency, we generate the x-direction gradient image and y-direction gradient image respectively. The BLB feature response is

$$f(BLB, t) = \sum_{i=1}^{8} sign(g(C_i, t) - g(C_0, t)) \times 2^{i-1} \tag{3.4}$$

$$g(C_i, t) = \begin{cases} \sum_{j \in C_i} I(j) & t = 0 \\ \sum_{j \in C_i} Gradx(j) & t = 1. \\ \sum_{j \in C_i} Grady(j) & t = 2 \end{cases}$$

$t$ is an indicator, $Gradx(j), Grady(j)$ are the gradient magnitude of pixel $j$ on x-direction and y-direction respectively.

The computation cost of BLB is similar to LBP and LAB because the sum of a region in both intensity image and gradient image could be calculated by integral images. Compared to traditional binary features, BLB achieves similar efficiency but better discriminative ability due to the using of variable patterns.

### 3.1.3 RealAdaBoost with BLB feature

We follow the RealAdaBoost introduced in Chapter 2.3.2, where each weak classifier is a pdf (probability distribution function) of the training samples. In consideration of the structure diversity, we add a structure-aware criterion into RealAdaBoost. The discriminative criterion of RealAdaBoost is Eq. 3.5

$$Z = 2\sum_{j} \sqrt{W_{+}^{j} W_{-}^{j}} + a \cdot fp \cdot D, \tag{3.5}$$

where the $W_{+}, W_{-}$ are internal parameters of RealAdaBoost, $D$ is the structure diversity of the features in Eq. 3.3, $a$ is the structure-aware factor to balance the discriminative capability and the feature diversity, $fp$ is the false positive rate of current stage. The Eq. 3.5 could be explained as follows: in the beginning stages of RealAdaBoost, the samples are still easy to be classified, so RealAdaBoost will refer to the features with more confident patterns. In the following stages, the training samples are complicated, then the features with diverse patterns might be utilized. This strategy makes sense, because the overall performance and robustness of a cascade boosted classifier are strongly influenced by the beginning stages which filter most of the candidate windows. In our experiment, we set the structure-aware factor a to 0.15. More details are given in Fig. 3.5.

### 3.1.4 Experiments

#### Frontal face detection

We train a face detector utilizing the proposed BLB feature and evaluate its performance on CMU + MIT frontal face dataset [87]. This dataset consists of 130 images containing 507 frontal faces with various conditions such as facial expression, occlusion, pose, and scale variations. Most images include more than one face on various backgrounds. 5-folds cross-validation is applied on this dataset to evaluate the proposed BLB feature. Fig. 3.6 plots the Receiver Operating Characteristics (ROC) curves of our method as well as other popular binary features and commonly-used face detection algorithms, including Haar [114], LBP [71], LAB [130], and heterogeneous features [73], in terms of the number of false positives with respect to the detection rate. As shown in Fig. 3.6, our detector achieves 96.0% detection rate at 0 false positive. To our knowledge, it is the best result for 0 false

```
Parameters:
  $N$     number of training samples
  $N_f$   number of randomly evaluated features in each iteration
  $T$     maximum number of weak classifiers
  $a$     penalty factor of the structure diversity
Input: Training set $\{(\mathbf{x}_i, y_i)\}, \mathbf{x}_i \in R^d, y_i \in \{-1, 1\}$
1. Initialize
   $w_i = \frac{1}{N}, H(\mathbf{x}_i) = 0, i = 1, \dots, N$
2. Repeat for $t = 1, 2, \dots, T$
  2.1 Update the sample weight $w_i$ using the $t^{th}$ weak classifier output $w_i = w_i e^{-y_i h_t(\mathbf{x}_i)}$
  2.2 For $m = 1$ to $N_f$
        For $n = 0$ to 2
    2.2.1 Randomly generate a BLB feature $F$ with structure diversify $D$
    2.2.2 Calculate the BLB response following Eq. 3.4
    2.2.3 Select the best BLB which minimizes Eq. 3.5
  2.3 Update weak classifier $h_t(x)$
  2.4 Update strong classifier $H_{t+1}(\mathbf{x}_i) = H_t(\mathbf{x}_i) + h_t(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^{T} h_j(\mathbf{x})]$
```

Figure 3.5: Selecting BBP feature using RealAdaBoost.

positive among the traditional features on the CMU + MIT frontal face dataset. Obviously, compared with other algorithms using single binary feature, the detection rate of our method is improved dramatically, especially for cases at low false alarms. The performance of our method is also similar to the classifier using heterogeneous features [73], which is trained by kernel SVM. In addition, we find that the boosted classifier with the penalty term (blue curve) on feature diversity achieves better accuracy than the one without penalty term (green curve), especially for lower false positive rate. The reason is that the features with larger structure diversity might be easily influenced by the noise, so that the corresponding classifier performs poor on object detection in real images.

### Pedestrian detection

We evaluate the proposed BLB feature using the INRIA pedestrian dataset [19]. The INRIA database contains 1,774 human annotations (3,548 with reflections) and 1,671 person free images. Detection on INRIA pedestrian dataset is challenging since it includes subjects with a wide range of variations in pose, clothing, illumination, background and partial occlusions. In the experiments, we firstly follow the training and testing protocols at patch level proposed by Dalal and Triggs [19]. In Fig. 3.7, we plot the miss rate tradeoff False Positive rate Per Window (FPPW) curves on a log-log scale by tuning the rejection threshold of the classifiers. We compare the BLB with the traditional HOG (Histogram of Oriented Gradient) [19], and the covariance matrices mapped to Riemann Manifold [109]. It can be seen that the BLB achieves 2% less miss rates at $10^{-4}$ FPPW comparing to HOG. The

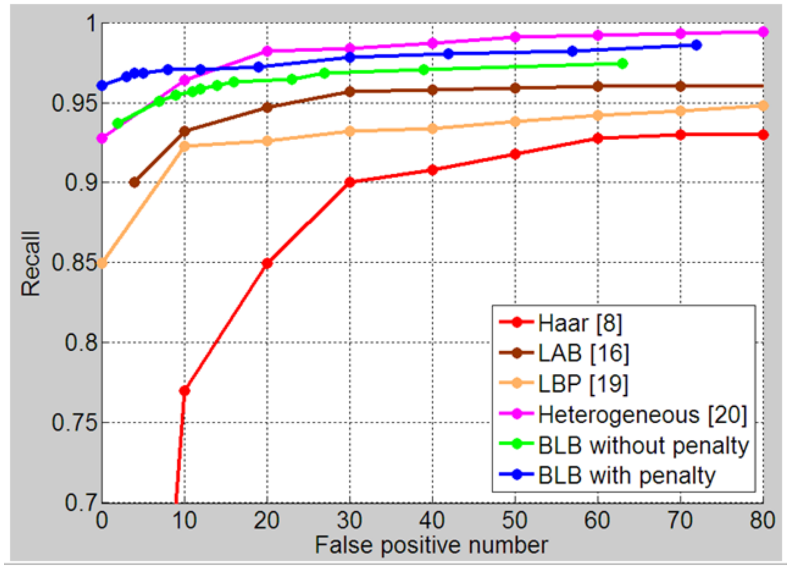


Figure 3.6: BLB experiments on CMU face dataset.

accuracy is also comparable with covariance matrix. But BLB's computation cost is much less compared to HOG and covariance matrix. In addition, the performance of the classifiers with penalty term (blue curve) is similar to the one without penalty (green curve) at lower FP. The reason is that FPPW evaluation utilizes patch windows, which is quite different from real detection.

Furthermore, we evaluate our method under the criteria of the detection rate versus False Positive rate Per Image (FPPI) [117]. Fig. 3.8 gives the comparison with the algorithms based on single feature [19][114][63] on INRIA dataset. Our algorithm shows competitive result with Haar [114], LBP, and HOG [19][63] feature.

### Side-view car detection

The side-view car detection performance is evaluated on the UIUC car dataset [2]. This dataset contains a single scale test set (170 images with 200 cars), a multi-scale test set (108 images with 139 cars), and a training set of 550 side-view car images. The car patches from the training images are resized to $100 \times 40$ pixels and horizontally flipped, so that there are totally 1,100 car patches in the positive training set. We also collect 1,000 images without any cars on the internet as the negative training set.

We compare our method with previous approaches following the Equal Precision and Recall rate (EPR) method. The results are listed in Table 3.1. It can be seen that the proposed method has high performance competitive to other methods on both single scale and multi-scale testing sets.

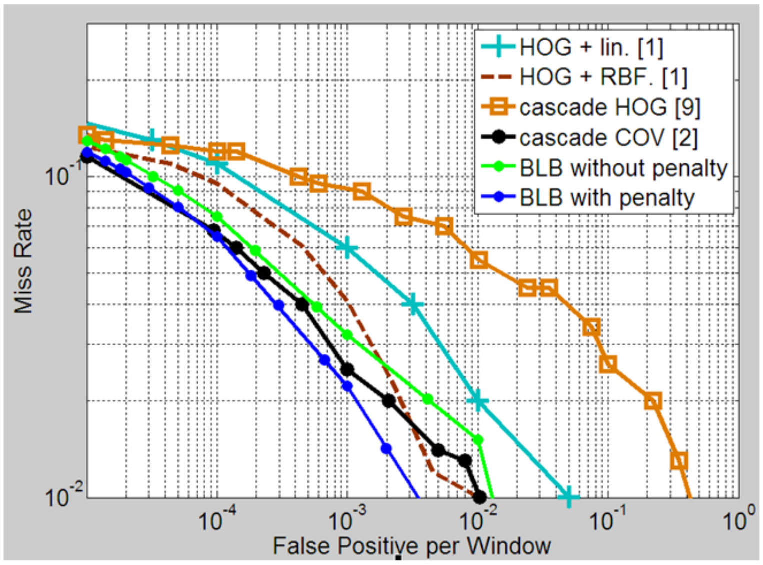


Figure 3.7: BLB FPPW evaluation on INRIA dataset.

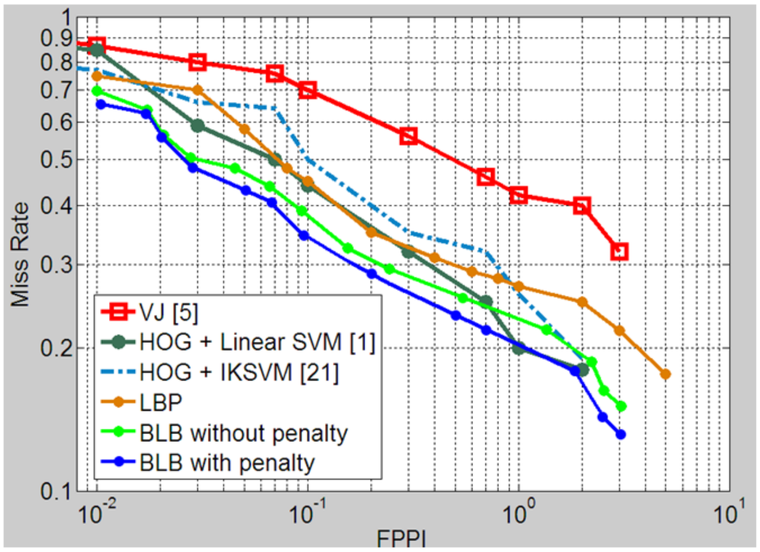


Figure 3.8: BLB FPPI evaluation on INRIA dataset.

Table 3.1: BLB experimental results (EPR) on UIUC car dataset.

| Method | Single-scale | Multi-scale |
|---|---|---|
| Leibe [57] | 0.975 | 0.95 |
| Lowe [67] | 0.999 | 0.906 |
| Wu [121] | 0.975 | 0.935 |
| Zheng [138] | 0.980 | 0.960 |
| Lampert [54] | 0.985 | **0.986** |
| BLB no penalty | 0.990 | **0.986** |
| BLB with penalty | **0.993** | **0.986** |

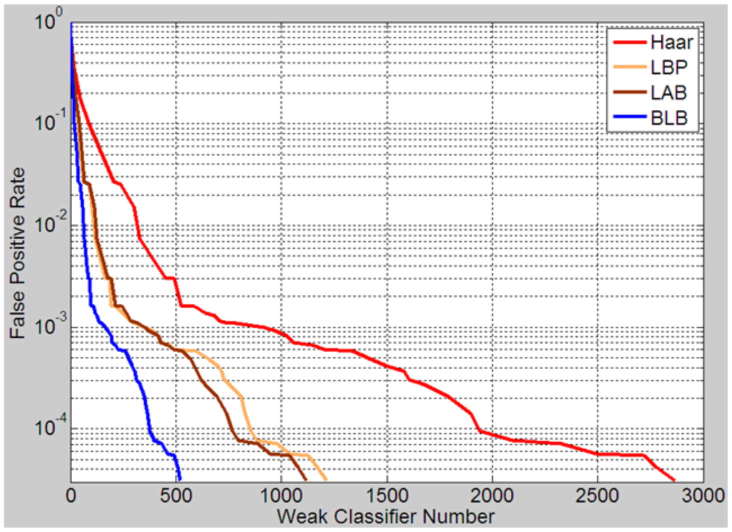


Figure 3.9: Convergence speed of the classifiers with different binary features on INRIA dataset.

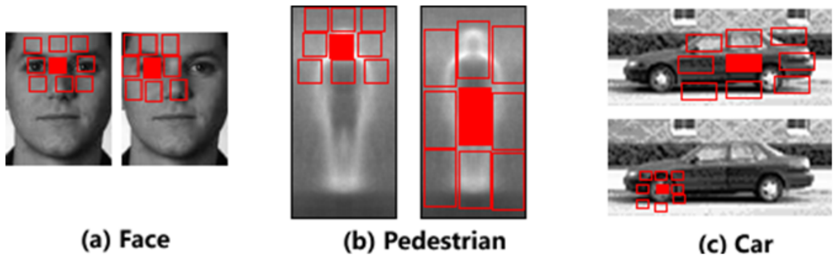


Figure 3.10: Selected first two BLBs of faces, pedestrians, and cars.

**Feature analysis**

We compare the convergence speed of the training process in INRIA pedestrian dataset. Fig. 3.9 plots the FPPW against the number of weak classifiers for different methods. The RealAdaBoost with penalty term on the feature diversity is utilized. This figure shows that BLB converges faster, at the rate of approximately two times faster than LAB and LBP. In addition, the performance of boosted classifier is shown to be positively proportional to the convergence speed in training. This signifies that the proposed BLB performs better on the training accuracy and the training speed of boosted classifiers. Moreover, we test the resulting face detector on a desktop PC with dual core 3.0GHz CPU and 8 GB memory. It only takes less than 10ms to detect all faces in a $640 \times 480$ input image if the minimum face size is $30 \times 30$.

Fig. 3.10 shows the first two selected BLB features of the faces, pedestrians and cars. These features could capture the typical structures of the objects, which might not be well covered by traditional binary features. For example, the two features in Fig. 3.10(c) reflect the car wheel and body pattern.

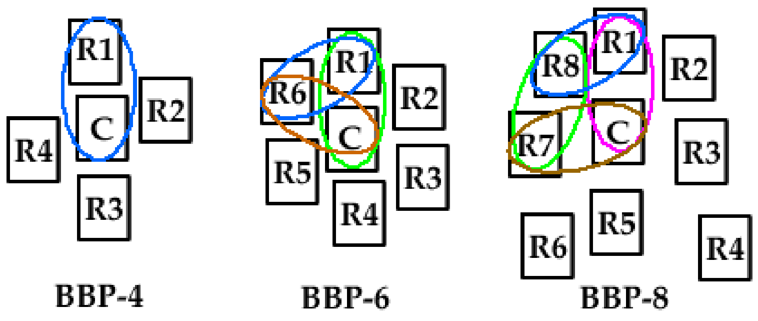


Figure 3.11: BBP features with adjacent pairs. Each ellipse shows an example of adjacent pair. There are 4 adjacent pairs in BBP-4, 12 in BBP-6, and 12 in BBP-8.

## 3.2 Boosted Binary Pattern

### 3.2.1 Boosted Binary Pattern (BBP)

Another important application of the binary feature is the object tracking. Since the tracking system considers the efficiency as an important factor as well as the accuracy, the binary features have great advantage compared to other local features. The above BLB feature achieves considerable accuracy for detection tasks. But for general object tracking, the key object characteristics might be beyond the description ability of the 8-bins binary coding in BLB. To solve this problem, we further generalized the BLB feature to the Boosted Binary Patterns (BBP), which is published in ICME 2015 [83]. Similarly, BBP comprises of the center rectangle $C$ and $n$ non-overlapped surrounding rectangles $R_1, R_2, \dots, R_n$. We denote it by BBP-n in Eq. 3.6, where the $(x_i, y_i)$ is the left-top corner of each rectangle, $(w_i, h_i)$ is the width and height

$$BBP_n = \{C, R_1, R_2, \dots, R_n\},\ R_i = \{x_i, y_i, w_i, h_i\},\ i = 1, \dots, n. \tag{3.6}$$

The surrounding rectangles $R_i$ are numbered from 1 to $n$, which starts at the top-middle one and increases in clockwise way. In our case, the $n$ could be 4, 6, or 8. Fig. 3.11 illustrates the examples for BBP-4, BBP-6, and BBP-8 respectively.

In BLB, it has been proved that the geometric relationship between rectangles pairs could contribute to the overall accuracy in the classification. So we define the *adjacent pairs* illustrated as the ellipses in Fig. 3.11. There are 4 adjacent pairs in BBP-4, 12 in BBP-6, and 12 in BBP-8. Similar to BLB, the overlap ratio of BBP-4, BBP-6, and BBP-8 is set to 0.5, 0.25, and 0.5 respectively. Fig. 3.12(a) illustrates a BBP-4 feature. In the adjacent pair $\{C, R_1\}$, $R_1$ should overlap at least 50% with $C$'s width, so the available region of $R_1$ restricted by $C$ is the cyan region bounded by the dot-dashed lines around $C$. Similarly, in Fig. 3.12(b), $R_1$ is included in three related adjacent pairs, $\{C, R_1\}, \{R_6, R_1\}, \{R_2, R_1\}$, so it should overlap at least 25% with $C$'s width, $R_2$'s height, and $R_6$'s height. These three

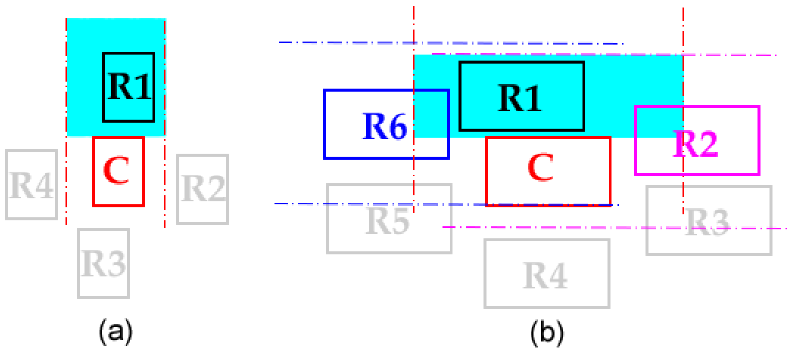


Figure 3.12: Constraint of the adjacent pairs. For BBP-4, the rectangles in adjacent pair should overlap at least 50%. For BBP-6, the rectangles in adjacent pair should overlap at least 25%.

constraints correspond to the dot-dashed lines around $C$, $R_2$ and $R_6$ respectively. As a result, the possible region of $R_1$ is the cyan region crossed by these dot-dashed lines.

Similarly, we use both the intensity image and the gradient image for BBP extraction. We still adopt the x-direction gradient image and y-direction gradient image instead of the true gradient. Given a BBP-n feature $BBP_n = \{C, R_1, R_2, \ldots, R_n\}$, the final feature response is given by

$$f(BBP_n, t) = \sum_{i=1}^{n} sign(g(R_i, t) - g(C, t)) \times 2^{i-1}. \tag{3.7}$$

$$g(R_i, t) = \begin{cases} \sum_{j \in R_i} I(j) & t = 0 \\ \sum_{j \in R_i} Gradx(j) & t = 1. \\ \sum_{j \in R_i} Grady(j) & t = 2 \end{cases}$$

The definations of $t, Gradx, Grady$ are the same as in Eq. 3.4. The structure diversity term of BBP is the same with BLB in Eq. 3.3.

### 3.2.2 Tracking with BBP

In the tracking procedure, we use RealAdaBoost to select the meaningful BBP features to describe the target frame by frame. Since the robustness is an important factor in object tracking, using confident features clearly reduces the possibility of losing target or false alarms. Thus it is better to consider the robustness for each feature in addition to the discriminative ability.

In each frame, the RealAdaBoost solves the classification task of several positive patches located on the target and negative patches around the target. We know that the margin of the weak classifier $h$ on $\mathbf{x}$ is $y \cdot h(\mathbf{x})$, where the $h$ is normalized to $[-1, 1]$. This margin

```
Input: Training set $\{(\mathbf{x}_i, y_i)\}, \mathbf{x}_i \in R^d, y_i \in \{-1, 1\}$
1. Initialize sample weight and classifier output
   $w_i = 0.01, H(\mathbf{x}_i) = 0$
2. Repeat for $t = 1, 2, \ldots, 10$
  2.1 Update the sample weight $w_i$ using the $t^{th}$ weak classifier output $w_i = w_i e^{-y_i h_t(\mathbf{x}_i)}$
  2.2 For $m = 1$ to 20
    2.2.1 Randomly generate a BBP feature
    2.2.2 Calculate the BBP response following Eq. 3.7
    2.2.3 Build the predict distribution function $W_\pm$
    2.2.4 Select the best BBP which minimizes the sum of Eq. 3.9 in
    three consecutive frames
  2.3 Update weak classifier $h_t(x)$
  2.4 Update strong classifier $H_{t+1}(\mathbf{x}_i) = H_t(\mathbf{x}_i) + h_t(\mathbf{x}_i)$
```

Figure 3.13: Selecting BBP feature using RealAdaBoost.

represents the classification ability. Larger margins imply lower generalization error and better robustness [94]. We define the normalized margin as

$$K(h, \mathbf{x}) = \frac{y \cdot h(\mathbf{x})}{k}, \tag{3.8}$$

where $\mathbf{x}$ is the sample vector, $h$ is the weak classifier, $y \in \{-1, 1\}$ is the class lable, $k$ is a parameter related with the structural diversity $D$ in Eq. 3.3, calculated as a sigmoid function

$$k = \begin{cases} 1 & D \leq 3 \\ 1.5 - \dfrac{1}{1 + \exp^{3-D}} & D > 3 \end{cases}.$$

The above equation means that we believe the BBP features with $D \leq 3$ are more confident, which include $LBP_{8,1}, LBP_{16,1}, LBP_{16,2}$, LAB, and BBPs whose average distance between the surrounding rectangles and the center rectangle is 3 times smaller than the rectangle size. For the BBPs with larger structural diversity, the confidence in the tracking will decrease. Integrating this term with the classification error $Z$ in the following feature selection protocol

$$Z = 2 \sum_j \sqrt{W_+^j W_-^j} - \frac{\alpha}{n} \sum_{i=1}^{n} K(h, \mathbf{x}_i). \tag{3.9}$$

In consideration of the information in time domain, we accumulatively calculate Eq. 3.9 in three consecutive frames after the initialization for the first and second frames. In each frame, we collect 50 positive and 50 negative samples around the target. 20 random BBP features are evaluated and 10 of them with minimum $Z$ will be selected to describe the target. The detail algorithm is illustrated in Fig. 3.13.

Table 3.2: Success rate SR(%) of different BBP features. Bold fonts indicate the best performance.

| Sequence | BBP-4 | BBP-6 | BBP-8 | BBP-ALL |
|---|---|---|---|---|
| Bolt | 77 | 79 | 82 | **85** |
| Biker | 59 | 64 | 74 | **76** |
| Cliff bar | 66 | 69 | 77 | **80** |
| David | 77 | 80 | 85 | **88** |
| Kitesurf | 60 | 68 | 75 | **82** |
| Occluded face | 86 | 89 | 96 | **100** |
| Skiing | 64 | 66 | 70 | **77** |
| Tiger | 63 | 64 | **68** | **68** |
| Twinings | 82 | 85 | 88 | **92** |
| Walking person | 84 | 82 | 88 | **92** |
| Average | 72 | 74 | 80 | **84** |

### 3.2.3 Experiments

We evaluate our tracking algorithm on 10 challenging publicly available sequences. These sequences are from [92], [136], and [6]. The Success Rate(SR) is utilized to evaluate the performance of our algorithm, which is calculated based on the following score

$$score = \frac{area(R_T \cap R_G)}{area(R_T \cup R_G)}, \tag{3.10}$$

where $R_T$ is the tracking bounding box and $R_G$ is the ground truth bounding box. The tracking result is considered as a success if the score is larger than 0.5.

Firstly, we compare the performance of different BBP features, including BBP-4, BBP-6, BBP-8, and the combination of all BBP features (BBP-All) using the algorithm described in Fig. 3.13. The single block size of BBP varies from $2 \times 2$ to one third of the size of tracking target. The tracking accuracy is given in Table. 3.2. It shows that the accuracy of the BBP feature increases along with the number of the surrounding rectangles. The BBP-8 consistently achieves better accuracy compared to BBP-6 and BBP-4. If we combine all BBP features together, the accuracy could be further improved. This result shows the effectiveness of variable binary patterns. It could also be seen that using the gradient information, the tracking accuracy is clearly improved for all BBP features. We know that in general object tracking, it is relatively difficult to classify the target and background only through the intensity information. The advantage of utilizing gradient information is clear.

In addition, we compare the performance of different binary features by integrating them into the same RealAdaBoost framework. To improve the accuracy, the Haar, LBP, and LAB are also applied on both the intensity domain and the gradient domain. For Haar and LAB feature, the single block size is the same as BBP. For LBP feature, the $LBP_{8,1}, LBP_{16,1}, LBP_{16,2}$ are integrated together. The first 6 columns in Table 3.3 show the experimental results. It could be seen that in all 10 sequences, BBP consistently achieves

Table 3.3: Success rate SR(%) in comparison with the traditional binary features and state-of-the-arts. Bold fonts indicate the best one. Italic fonts indicate the second best.

| Sequence | Haar | LBP | LAB | BBP | CT [136] | MIL [6] | OAB [36] | TLD [48] |
|---|---|---|---|---|---|---|---|---|
| Bolt | 71 | 76 | 73 | **85** | 79 | *83* | 61 | 41 |
| Biker | 59 | 69 | 62 | **76** | *75* | 66 | 42 | 42 |
| Cliff bar | 64 | 67 | 71 | *80* | **89** | 65 | 23 | 77 |
| David | 78 | 59 | 66 | 88 | *89* | 68 | 31 | **98** |
| Kitesurf | 47 | 44 | 55 | *82* | 68 | **90** | 31 | 65 |
| Occluded face | *96* | 94 | 90 | **100** | **100** | 99 | 47 | 56 |
| Skiing | 70 | *71* | 68 | **77** | 70 | 62 | 69 | 59 |
| Tiger | 60 | 59 | 65 | **68** | 60 | 55 | 37 | 41 |
| Twinings | 90 | 77 | 88 | *92* | 89 | 72 | **98** | 46 |
| Walking person | 86 | 65 | 67 | **92** | *89* | 62 | 86 | 60 |
| Average | 72 | 74 | 73 | **84** | *81* | 72 | 51 | 59 |

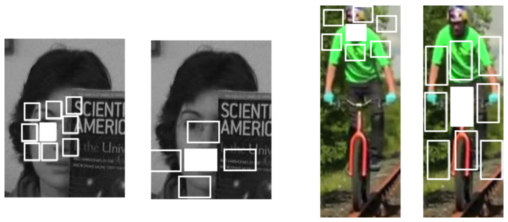


Figure 3.14: The first two selected BBP features of faces and bikers.

at least 5% better accuracy compared to Haar, LBP, and LAB. This signifies the advantage of variable binary patterns. We also notice that using the penalty term of robustness, the average accuracy could be further improved 4%, especially for those targets with large pose variant such as bolt, kitesurf, and skiing. This implies the importance of balancing the discriminative ability and robustness in the tracking system.

Moreover, we compare our method with commonly-used algorithms, including compressive tracking [136], online AdaBoost (OAB) [36], MIL [6], and TLD [48]. These algorithms achieve the state-of-the-art results in these sequences. Since all of these trackers involve randomness, we run them 5 times and report the average result for each video clip. The last four columns in Table 3.3 give these quantitative results. It can be seen that the proposed algorithm achieves the best or second best results in most sequences in terms of the success rate. We find that if the tracking target has clear region contrast or rigid shape, the tracking accuracy will be better. As shown in Fig. 3.14, in the "Occludedface" sequence, the eye and mouth region show a clear contrast to the surrounding regions. In the "Biker" sequence, the body contour of the biker is also a clear feature. These typical structures are well captured by the first select two BBPs. We test our tracker on a Intel Dual-Core 3.0 GHz PC with 8GB memory. It runs at 30+ frames per second (FPS), which also shows comparable efficiency with the state-of-the-art methods.

## 3.3 Conlcusion

In this chapter, we give a brief introduction of the commonly-used binary features LBP and LAB. Although these features has good efficiency, their discriminative ability is relatively low due to the limited binary patterns. To solve this problem, we build a large binary feature pool with variable-location and variable-size blocks for both the intensity domain and the gradient domain, called Boosted Local Binaries. We further utilize RealAdaBoost algorithm to select informative features by evaluating different blocks pairs while considering the structure diversities. Experimental results show that the proposed BLB achieves high accuracy on face, pedestrian and car detection tasks. It also speeds up the convergence compared to standard binary features.

Although BLB shows promising results on some object categories, it does not work well for general object tracking. We further design Boosted Binary Patterns with more flexible patterns. A boosting framework which is capable of balancing the discriminative ability and robustness of the binary features is further proposed to online update the tracking model. Experimental results show that BBP achieves very good results in the public challenging videos.

# Chapter 4

# Mid-order features

Mid-order features utilize the gradient information to reflect specific characteristic. The commonly-used mid-order features include the histogram features such as SIFT [61], HOG [19], and the wavelets such as Gabor. In our research, we propose two mid-order features, the Edge Histogram of Oriented Gradient (Edge-HOG), and the Deformable Contour Feature (DEF).

## 4.1 Edge Histogram of Oriented Gradient

Histogram of Oriented Gradient (HOG) [19] is one of the most commonly-used gradient features in object detection and recognition. HOG breaks the image region into a cell-block structure and generates histogram based on the gradient orientation and spatial location. The input region (block) is first divided into small connected regions, called cells, and for each cell a histogram of edge orientation is computed. The histogram channels are evenly spread from 0 to 180 degrees. Furthermore, the histogram counts are normalized for illumination compensation. This can be done by accumulating a measure of local histogram energy over the somewhat larger connected regions and using the results to normalize all cells in the block. The combination of these histograms represents the final HOG feature. A figure of the HOG extraction is shown in Fig. 4.1.

HOG ignores some important structural information in each cell. If two cells contain the same edge but at different positions, the resulting feature vectors will be the same. To solve this issue, we propose a enhanced version of HOG feature named Edge-HOG in ICIP 2014 [82]. Edge-HOG arranges the cells along an edge template to gain additional structure information. In addition, we extract a complementary feature vector based on the gravity centers, so that it captures more discriminative information than the traditional HOG does.

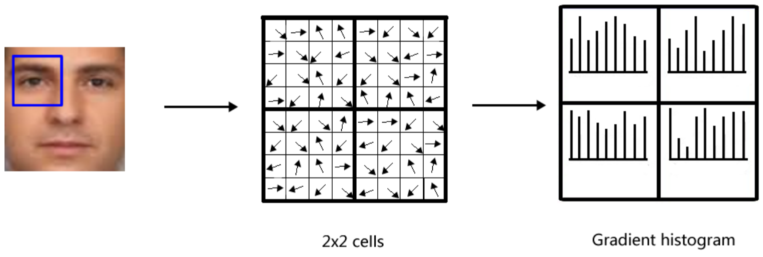


Figure 4.1: Traditional HOG feature extraction.

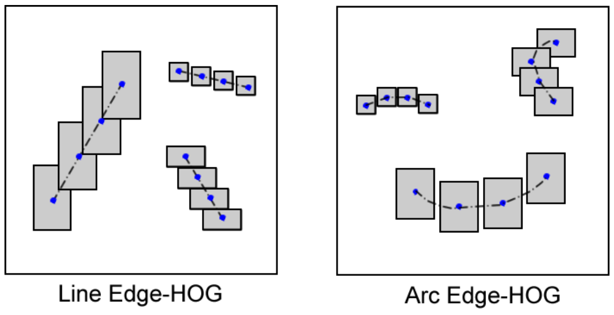


Figure 4.2: Edge-HOG features.

### 4.1.1 Edge Histogram of Oriented Gradient (Edge-HOG)

An Edge-HOG feature consists of a series of cells along an edge template, as shown in Fig. 4.2. The edge template (dot-dashed lines) can be lines and arcs in variable length, positions and directions. 4 pixels (blue dots) are uniformly sampled on these edges as the center pixels of Edge-HOG cells. These cells could overlap or lay far away from each other. To reduce the size of the feature pool, we add a constraint on the distance of the two neighboring center pixels $d$ and the cell size $(w, h)$ as $0.75max(w, h) < d < 2min(w, h)$. Since the cells are arranged along an edge template, the Edge-HOG not only extracts the statistical information of the local region, but also reflects the edge position and direction.

In Edge-HOG, each bin of the gradient vector is the weighted sum of all pixels with the same gradient orientation in a cell. It can be readily extracted using the integral image algorithm. Besides the cell arrangement, we induce some structure information into the feature extraction to further improve the discriminative ability of Edge-HOG. An 8-bins structure vector for each cell in the Edge-HOG feature will be calculated based on the geometric information. We first divide the cell into 8 regions according to angle of each pixel and the cell center. For each region, the gravity centers are calculated, and the distance of the gravity center and the cell center is used to generate the structure vector. As shown in Fig. 4.3, the cell is divided into 8 regions numbered from 0 to 7. The $r_i$ illustrates the Euclidean distance between the gravity center of $i$th region and the cell center, which will be used in the structure vector extraction. Denote the coordinate of cell center by $(x_c, y_c)$, the $i$th bin of the structure vector $\mathbf{d}_{grav}$ is calculated as Eq. 4.1

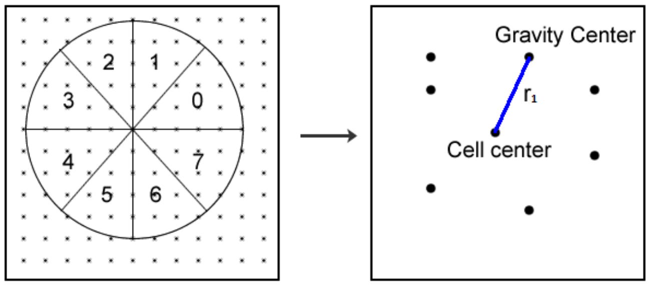


Figure 4.3: Edge-HOG feature extraction.

$$\mathbf{d}_{grav,i} = \frac{\sum_{(x,y)\in R_i} r_i \times grad_{x,y}}{\sum_{(x,y)\in R_i} grad_{x,y}}, \tag{4.1}$$

where $R_i$ is one of the 8 regions of each cell in Edge-HOG, $grad$ is the gradient magnitude. L1-normalization is applied on all cells after the feature extraction.

### 4.1.2 RealAdaBoost with Edge-HOG

We adopt RealAdaBoost to train the detectors based on Edge-HOG. In each iteration, the candidate Edge-HOG features are evaluated on all positive and negative samples. To learn the best weak classifiers, the most intuitive way is to look through the whole feature pool, which is rather time consuming. So we resort to a sampling method to speed up the feature selection process. More specifically, a random sub-sample of size $log 0.05/log 0.95 = 59$ will guarantee that we can find the best 5% features with a probability of 95%. For each candidate block, the feature vectors are extracted and further used to train a linear classification plane using least squares. Then the final feature value used to build the probability distribution as the weak classifier. Fig. 4.4 gives more details.

### 4.1.3 Experiments

#### Experiments on INRIA pedestrian dataset

We first evaluate the proposed Edge-HOG feature using the INRIA pedestrian dataset. Multi-scale Edge-HOG features from $8\times 8$ to $32\times 32$ cell sizes are utilized to train a cascade classifier for the $64 \times 128$ scanning window. 6,738 Edge-HOG features are generated.

In Fig. 4.5(a), we plot the miss rate tradeoff False Positive rate Per Window (FPPW) curves. We first compare the performances of the boosted classifiers with different cell arrangements and feature vectors. It can be seen that the Edge-HOG (green curve) shows better results compared to the traditional HOG [140] using the same gradient feature vector and AdaBoost training algorithm (red curve). If we use the Edge-HOG with the proposed structure vector, the accuracy (blue curve) is further improved. We also compare our method with three commonly-used methods, HOG with Linear SVM [19], HOG with Kernel

```
Parameters
  N     number of training samples
  M     number of evaluated features each iteration
  T     maximum number of weak classifiers
Input: Training set $\{(\mathbf{x}_i, y_i)\}, \mathbf{x}_i \in R^d, y_i \in \{-1, 1\}$
1. Initialize sample weight and classifier output
   $w_i = 1/N, H(\mathbf{x}_i) = 0$
2. Repeat for $t = 1, 2, \dots, T$
  2.1 Update the sample weight $w_i$ using the $h^{th}$ weak classifier output $w_i = w_i e^{-y_i h_t(\mathbf{x}_i)}$
  2.2 For $m = 1$ to M
    2.2.1 Extract Edge-HOG features vectors
    2.2.2 Train a classify plane to map the Edge-HOG vector to $\{-1, 1\}$
    2.2.3 Select the best feature with the lowest classification error
  2.3 Update weak classifier $h_t(x)$
  2.4 Update strong classifier $H_{t+1}(\mathbf{x}_i) = H_t(\mathbf{x}_i) + h_t(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^{T} h_j(\mathbf{x})]$
```

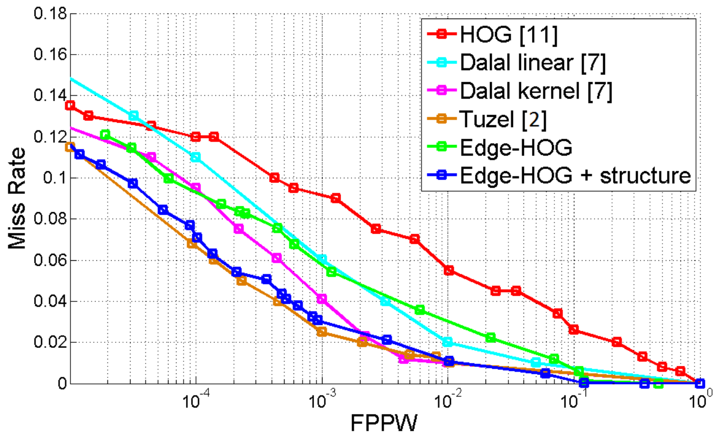
Figure 4.4: Learning the Edge-HOG features using RealAdaBoost.

SVM [63], and the covariance matrix with AdaBoost [109]. It can be seen that the proposed Edge-HOG method achieves detection rate of 92.9% at FPPW $10^{-4}$, which is similar to the covariance matrix. But the computation cost is much lower.

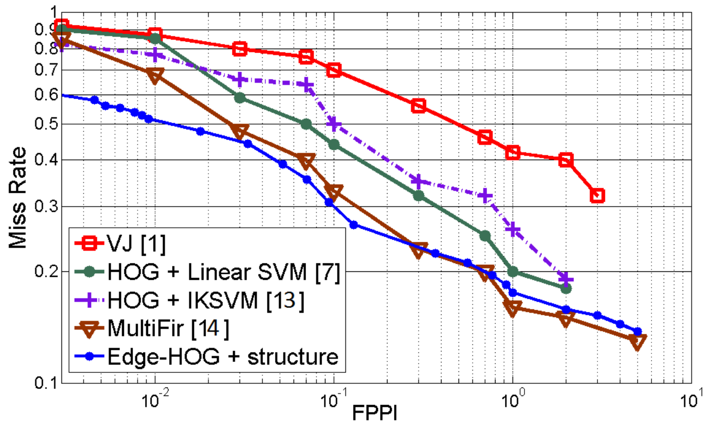
Furthermore, we evaluate our method under the criteria of the detection rate versus False Positive rate Per Image (FPPI). Fig. 4.5(b) shows that our algorithm also achieves competitive result with Haar [114] and HOG [19]q[63]. The accuracy is similar to the multi-feature combination [120].

Next, we compare the convergence speed of the training process in INRIA pedestrian dataset. Fig. 4.6 plots the FPPW against the number of weak classifiers for different methods. This figure shows that the Edge-HOG converges faster, at the rate of approximately two times faster than the HOG. When we utilize the proposed structure vector, the convergence speed is further improved. In addition, the performance of boosted classifiers is shown to be positively proportional to the convergence speed in training. This signifies that the Edge-HOG performs better on the training accuracy and speed of boosted classifiers compared to the traditional HOG.

Moreover, we investigate how the Edge-HOG captures the structure information. We show the first 5 selected features in the boosting training in Fig. 4.7. The black rectangles in each picture represent the cells in one Edge-HOG feature. The bright background reflects the average profile of a pedestrian. From the figure, it can be seen that the first 5 selected features are basically along the profile of the pedestrian, which shows that the proposed Edge-HOG feature is able to describe the structure of the human body.



(a) FPPW evaluation on INRIA dataset



(b) FPPI evaluation on INRIA dataset

Figure 4.5: Edge-HOG accuracy evaluation on INRIA pedestrian dataset.

**Edge-HOG experimental results (EPR) on UIUC car dataset**

We evaluate our algorithm on UIUC car dataset. The car patches from the training images are resized to $64 \times 32$ pixels and horizontally flipped. We also collect 10,000 images without any cars from the internet as the negative training set. 1,764 Edge-HOG features are generated for $64 \times 32$ window.

We compare our approach with previous approaches following the Equal Precision and Recall (EPR) rate method. The results are listed in Table 4.1. It can be seen that our algorithm achieves competitive performance to other methods on both single scale and multi-scale test sets.

## 4.2 Deformable Contour Feature

The local shape features are relatively consistent to illumination variance, occlusions and background noises, so that they are able to classify the target object with the background [97]. In the ideal case, the local shape features should not only describe the object contour,

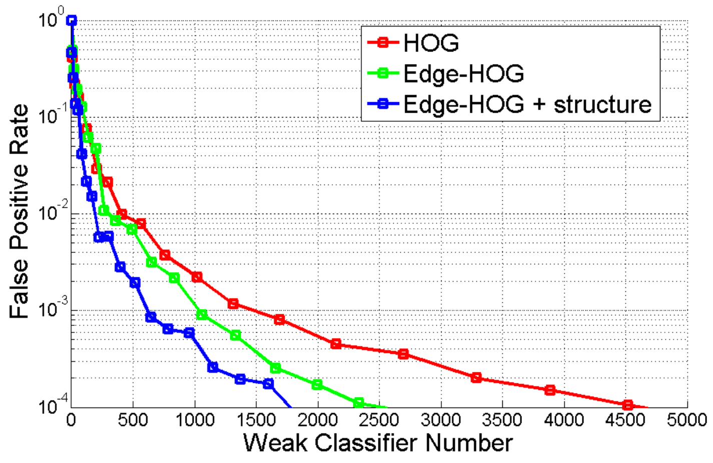


Figure 4.6: Convergence speed of the boosted detectors base on Edge-HOG in INRIA pedestrian dataset.

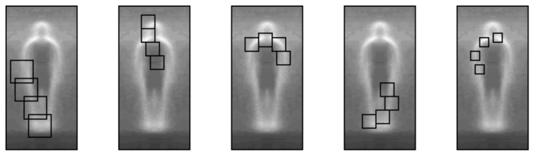


Figure 4.7: The first 5 selected Edge-HOG features.

but also classify the background edges with the foreground edges. Most of the traditional deformable edge features solve the matching problem in a local region, so that it may mismatch to inner contours or background edges. In addition, the geometric relationship between the deformable edge features is also important. Grouping the deformable edges has certain advantage because it is consistent with the perceptual grouping and recognition of human vision. It will also contribute to the description ability of continuous contours.

Inspired by these issues, we propose the Deformable Edge Set (DES) in ICIP 2015 [84]. The key idea is to encode the local shape using deformable templates and then to group them to capture higher level shape characteristics. Our contributions are two folds. Firstly, the Deformable Edge Feature (DEF) is designed based on a set of edge templates and the

Table 4.1: Experimental results on UIUC car dataset.

| **Approach** | **single scale** | **multi-scale** |
|---|---|---|
| Saberian et al. [91] | 99.0% | 92.1% |
| Xu et al. [127] | 99.5% | 98% |
| Wu et al. [121] | 97.5% | 93.5% |
| Lampert et al. [54] | 98.5% | 98.6% |
| Karlinsky et al. [49] | 99.5% | 98.0% |
| Edge-HOG | 99.5% | 98.6% |

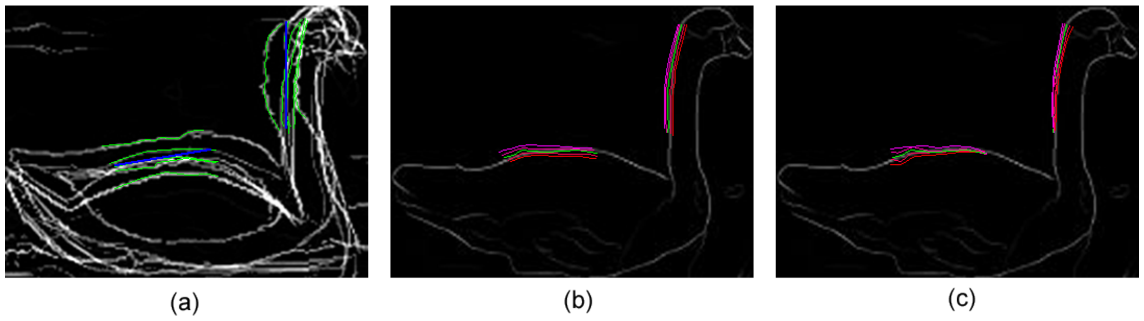


Figure 4.8: Illustration of DEF. (a) Edge template (blue lines) and matched edges (green lines) in 5 swans. (b) and (c) show the deformation of the first and second principal components (the eigenvalue increases from pink to red lines).

distribution model of pixels [30]. In addition, the DES is constructed to encode the geometric relationship between neighboring DEFs. This process is achieved by efficiently learning the DEF arrangement and model parameters in a subspace. Compared to traditional shape features, the DES focuses on the continuous contours rather than other shape characteristics.

### 4.2.1 Deformable Edges

In this section, we will introduce how to represent the local shape information by DEF. DEF extraction consists of two steps, template matching and edge deformation. In the template matching procedure, we generate a set of lines from 6 pixels to 1/3 of the object window size as the edge templates. Given an input edge image $E$, each edge template $t$ is matched to the edges in $E$ to get the best matching result $e^*$ following

$$e^* = \underset{|e|=|t|, e\in E}{argmin}\ D(e,t), \tag{4.2}$$

where $|e|$ is the length of edge $e$, $D(e,t)$ is the normalized distance between two edges $e$ and $t$ with the same length

$$D(e,t) = \frac{1}{|e|}\sum_{i=1}^{|e|}(d(e_i,t_i)+\alpha|O(e_i)-O(t_i)|). \tag{4.3}$$

In Eq. 4.3, $e_i$ and $t_i$ are the $ith$ pixel of $e$ and $t$ respectively, $d(e_i,t_i)$ is the Euclidean distance between these two pixels, $O(\cdot)$ is the normalized orientation, and $\alpha$ is the constant to balance the weight of the geometric location and the orientation. Fig. 4.8(a) shows two templates and the matched contours in 5 swan images of ETHZ database.

In the edge deformable procedure, the distribution model of the pixels in the matched edges is utilized. For each edge template, denote the matched edge $e^*$ in an image by a $2|e^*|$ dimensional vector based on its coordinate $\{e^*_{1,x}, e^*_{1,y}e^*_{2,x}, e^*_{2,y}, \ldots\}$, the DEF $f$ is generated by applying PCA on all training images

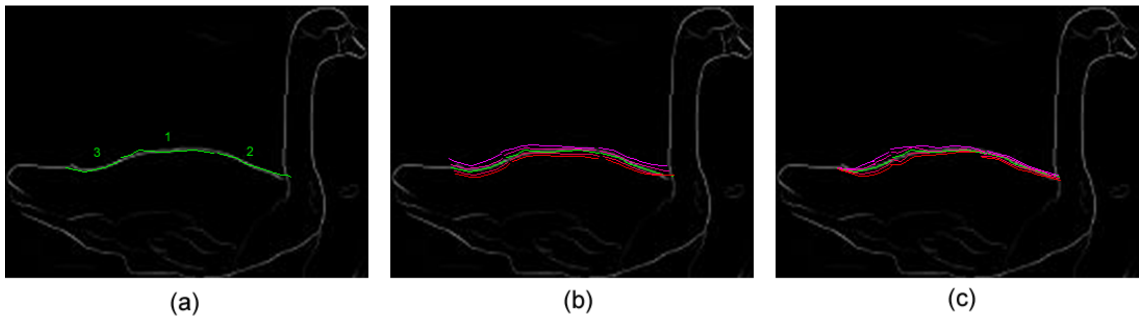


Figure 4.9: Illustration of DES. (a) DES which consists of 3 DEFs (green lines). (b) and (c) show the deformation of the first and second principal components (the eigenvalue increases from pink to red lines).

$$f_{a_1,\ldots,a_p} = \mu + \sum_{i=1}^{p} a_i v_i, \tag{4.4}$$

where $\mu$ is the mean edge over all samples, $v$ is the eigenvector, the parameter $a_i$ is bounded by $|a_i| \leq 2\sqrt{\lambda_i}$, and $\lambda$ is the eigenvalue. The Eq. 4.4 means that DEF encodes the deformation of edges by $v$ with the weight $a_i$. Fig. 4.8(b) and Fig. 4.8(c) show the deformation described by the first two components. It could be seen that the edge deforms from pink lines to red lines, along with the increasing of $a_i$.

In our implementation, we utilize the first three components so that it allows us to keep at least 95% energy. Given an input edge graph $E$ and a DEF $f_{a_1,\ldots,a_p}$, the matching cost is calculated by

$$C(E, f) = \min_{i,j,k,e \in E} D(e, f_{a_i,a_j,a_k}). \tag{4.5}$$

We quantize the $a_i, a_j, a_k$ to 15 bins, 10 bins, and 6 bins respectively. The distance transform [97] is utilized to calculate the optimal $a_i^*, a_j^*, a_k^*$ efficiently.

### 4.2.2 Constructing Deformable Edge Set (DES)

The Deformable Edge Set (DES) is defined as a set of DEF

$$F = \{f^1, f^2, \ldots, f^n\}, \tag{4.6}$$

where the DEFs in DES are arranged in a chain to describe the continuous contour. Each $f^i, 1 < i < n$ is adjacent to $f^{i-1}$ and $f^{i+1}$, which leads to two constraints. Firstly, the two adjacent DEFs should not lay far away from each other, so one vertex of these DEFs need to be close enough. In addition, to describe the continuous contour, the other vertex of these two DEFs should not be too close. These constraints are formulated as

$$dis(\mu^i_{|\mu_i|}, \mu^{i+1}_1) \leq 6 \quad dis(\mu^i_1, \mu^{i+1}_{|\mu_{i+1}|}) \geq \frac{1}{4}(|\mu^i| + |\mu^{i+1}|), \tag{4.7}$$

where $\mu^i$ is the mean of $f^i$ over all images, $\mu^i_j$ is the $j$th pixel of $\mu^i$, $|\mu|$ is the length of $\mu$, $dis$ is the Euclidean distance.

Given an edge map $E$ and a DES $F$, the matching cost of DES is the sum of the matching cost of each DEF in Eq. 4.5 and the cost of all adjacent DEFs

$$C(E, F) = \sum_{i=1}^{n} C(E, f^i) + \beta \sum_{i=1}^{n-1} C(f^i, f^{i+1}), \tag{4.8}$$

where

$$C(f^i, f^{i+1}) = |D(f^i, \mu^i) - D(f^{i+1}, \mu^{i+1})|, \tag{4.9}$$

$\beta$ is the weight of the adjacent cost, set as 1.25 in our experiments. The adjacent cost in Eq. 4.9 is calculated by accumulating the deformation differences of two adjacent DEFs. This cost will be small in a well-matched object since the adjacent edges in any objects has similar deformations. Fig. 4.9(a) gives the example of a DES which consists of 3 DEFs.

### 4.2.3 Using DES for recognition

Because we utilize 3 components in DEF extraction, for a DES $F = \{f^1, f^2, \ldots, f^n\}$, minimizing Eq. 4.8 requires us to solve the optimization problem with $3n$ parameters. Using brute force to enumerating the possible solutions is not realistic. Then we consider the $3n-$dimensional parameter space $M = \{a_{1,i}, a_{1,j}, a_{1,k} \ldots a_{n,i}, a_{n,j}, a_{n,k}\}$. Similar to the strategy use in Chapter 4.2.1, we move the optimization problem to the subspace by PCA. The model parameters of DES are estimated by

$$M_{b_1,\ldots,b_p} = \mu' + \sum_{i=1}^{p} b_i v'_i, \tag{4.10}$$

where $\mu'$ is the average parameters for all training samples, $v'_i$ is the eigenvector, the parameter $b_i$ is bounded by $|b_i| \leq 2\sqrt{\lambda'_i}$, and $\lambda'$ is the eigenvalue. We still use the top three principal components $b_i, b_j, b_k$ and quantize them to 15 bins, 10 bins, and 6 bins respectively. As a result, the learning process of DES parameters takes $O(15 \times 10 \times 6)$ time, which is far more efficient than the brute force strategy.

In the training process, we begin with an initial DEF and incrementally grow it into a DES. In each iteration, we search all adjacent DEFs following the constraint in Eq. 4.7. For each candidate DEF, the matching cost function of adding it to current DES (Eq. 4.8) is updated, and the model parameter $M$ is calculated to update the parameter subspace in Eq. 4.10. Then the DEF with the minimum cost will be added to current DES. This process is repeated until the subspace fail to keep 95% of the energy or the total number

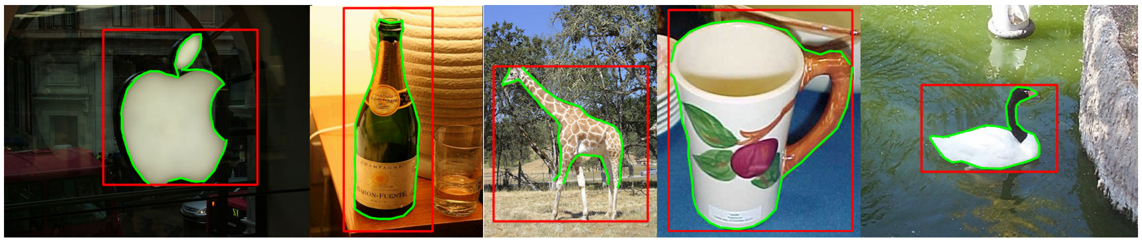


Figure 4.10: Examples of object and contour location result in the ETHZ database.

Table 4.2: Comparison of Average Precision (AP) with commonly-used algorithms on ETHZ shape database.

| | Felz.[28] | Ma [62] | Wang [116] | Srinivasan [100] | Li [59] | DEF | DES |
|---|---|---|---|---|---|---|---|
| Applelogos | 89.1 | 88.1 | 88.6 | 84.5 | 82.3 | 90.1 | **92.0** |
| Bottles | 95.0 | 92.0 | **97.5** | 91.6 | 90.0 | 92.0 | **97.5** |
| Giraffes | 60.8 | 75.6 | **83.2** | 78.7 | 69.2 | 80.4 | 81.8 |
| Mugs | 72.1 | 86.8 | 84.3 | 88.8 | **98.0** | 86.8 | 89.8 |
| Swans | 39.1 | **95.9** | 82.8 | 92.2 | 81.0 | 88.2 | 92.6 |
| Mean | 71.2 | 87.7 | 87.3 | 87.2 | 84.1 | 87.5 | **90.2** |

of DEF exceeds 12. Fig. 4.9(b) and Fig. 4.9(c) illustrate a DES deformed by the first and second principal component.

We utilize DEF and DES as shape features for object recognition in RealAdaBoost framework. The minimum matching cost in Eq. 4.7 is utilized as the feature response. The objects are detected using the classifier trained to $10^{-6}$ false positive rate. After the detection, the object contours will be located in the detected bounding box. Given a pixel $x$ in a detection window, the probability of $x$ being in any edges is the sum of the confidence of DEFs $f$ including $x$

$$P(x) = \sum_{x \in f^i} \frac{W^+_{C_x}}{W^+_{C_x} + W^-_{C_x}}, \tag{4.11}$$

where $C_x$ is the matching score corresponding to $f^i$, $W^{\pm}$ is the distribution on positive samples and negative samples respectively. The final contours are obtained by averaging Eq. 4.11 in the sliding windows of multiple scales.

### 4.2.4 Experiments

We show the performance of the proposed method using the ETHZ shape dataset [29], which consists of 5 classes including applelogo, bottle, giraffe, mug and swan. This dataset is challenging since the objects appear in a wide range of scales with intra-class shape variations. We follow the training and testing protocol in [29]. For applelogos and giraffes, $80 \times 80$ windows with 6,154 edge templates are utilized. For mugs, $100 \times 90$ windows with 6,982 edge templates are adopted. $100 \times 60$ windows and 5,610 edge templates are used for

Table 4.3: Accuracy of localized object boundaries. Each entry is the AC/AP.

| | Bounding boxes [30] | Ferrari [30] | DEF | DES |
|---|---|---|---|---|
| Applelogos | 42.5/40.8 | 91.6/**93.9** | 91.4/91.5 | **92.7**/93.2 |
| Bottles | 71.2/67.7 | 83.6/84.5 | 83.7/83.1 | **84.5**/**87.2** |
| Giraffes | 26.7/29.8 | 68.5/77.3 | 65.0/75.2 | **73.3**/**79.2** |
| Mugs | 55.1/62.3 | **84.4**/77.6 | 82.7/77.6 | 83.9/**80.1** |
| Swans | 36.8/39.3 | 77.7/77.2 | 74.3/75.2 | **80.2**/**84.4** |

swans, while $40 \times 100$ windows and 2,772 edge templates for bottles. The $\alpha$ in Eq. 4.3 are set to 4 for applelogos, swans and giraffes, 6 for mugs, and 3 for bottles.

In Table 4.2, we compare the Average Precision (AP) of our algorithm with the commonly-used algorithms [116][100][28][62][59]. These algorithms achieve the best results among the non-deep learning methods on this database. It could be seen that DEF shows comparable result with these algorithms. The DES achieves about 3% better accuracy compared to DEF, which also shows the best result on 2 categories, and the second best results on the other 3 categories. This signifies the advantage of grouping DEF to capture the continuous contour information.

In Table 4.3, we compare the performance of the boundary location with [30] using both the AP and the Average Coverage (AC) as the evaluation protocol. It shows that only using DEF, the accuracy is slightly lower than [30]. Combining DEF to DES, the accuracy is significantly improved, especially for the swans and giraffes. This result is reasonable since compared to DEF, DES is able to filter some false matches on the inner or background edges. For the swans images which include a lot of background edges in the water surface, or the giraffes images where the giraffes are surrounded by the forest and prairie, the advantage of using DES is clear. Fig. 4.10 shows some recognition examples and localized contours.

## 4.3 Conclusion

In this chapter, we introduce two works of the mid-level features. The Edge Histogram of Oriented Gradient arranges the blocks along an edge to combine the geometric information and gradient statistic. In addition, a new feature extraction method is proposed based on the local structural information, as complementary to the traditional gradient histogram. Experimental results show that the Edge-HOG performs better on both the accuracy and training efficiency compared to traditional HOG feature.

Another work is the Deformable Edge Set. DES consists of several deformable edge features (DEF). The local shape information is encoded by the pixel distribution model in each DEF. They are further grouped based on the geometric relationship to describe continuous contour by DES. All the deformable parameters are learned in the subspace. The experimental results show that the boosted classifiers trained on DEF and DES work well on both the object detection and boundary localization tasks.

# Chapter 5

# High-order features

High-order features extract more complicated information as feature vector, such as second-order gradient, or the covariance matrix of low-level information. Recently, the co-occurrence features have become a hot topic. The co-occurrence information extracted by these features are able to capture some complicate object characteristics. Unfortunately, they also lead to heavy computation cost because the dense feature vector is time-consuming to calculate. In addition, we know that the performance of an object detector is decided not only by the discriminative ability of the features, but also by their *generalization power*, which is defined as the ability to deal with the cases that are not part of the training process. Due to the fact that some diverse co-occurrence patterns are sensitive to background noise, most of the co-occurrence features have poor generalization power. The detector succeeding in one scene might fail in another scene with different conditions, such as pose, illumination, etc. This is actually a trade-off problem. Although using stronger features may contribute to the training accuracy, it will increase the risk of both low generalization power and high computation cost.

To solve this problem, we design a set of localized co-occurrence features which can be computed efficiently. We generalize a kind of *co-occurrence patterns* that could be applied on any low-level features. Three kinds of co-occurrence features, CoHaar, CoLBP, and CoHOG are constructed as examples. In addition, the proposed co-occurrence features could be easily integrated with a Generalization and Efficiency Balance framework (details in Chapter 7.1) to further improve the accuracy and robustness. With the experiments on PASCAL VOC dataset, the proposed co-occurrence features are proved to be one of the most discriminative traditional features. This research is part of our ICCV paper [85].

## 5.1 Co-occurrence patterns

The co-occurrence features can be constructed based on the statistics information of several pre-defined *co-occurrence patterns*. Each co-occurrence pattern $\{U, V, F_1, F_2\}$ is a constraint

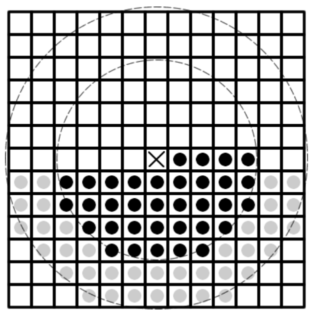


Figure 5.1: The pixel pairs in co-occurrence patterns. Each black pixel and center pixel correspond to a pixel pair. The highlight parts show the pairs with offsets $U \leq 4, V \leq 4$.

on a pixel pair $a = \{x_1, y_1, f_1\}$ and $b = \{x_2, y_2, f_2\}$. $(a, b)$ should satisfy the following condition

$$|x_1 - x_2| = U, y_1 - y_2 = V, f_1 = F_1, f_2 = F_2. \tag{5.1}$$

In Eq. 5.1, the $(x_1, y_1), (x_2, y_2)$ are the coordinates of $a$ and $b$. The *offset* $U, V \geq 0$ show the spatial distance of pixel $a$ and $b$. $f_1, f_2$ are scores of $a$ and $b$ generated by feature extraction algorithms. $F_1, F_2$ are constants in the score space $F$. As shown in Fig. 5.1, each black pixel and the center pixel correspond to the pixel pair of a co-occurrence pattern with $U \leq 4, V \leq 4$.

To compute a stable distribution that is robust against noise, we utilize the histogram based on the division of score space $F$ as the co-occurrence feature vector. Given an input window $R$, the offset $U, V$, and an extraction method to generate $F$, we divide $F$ into $n$ bins $\{F_1, \ldots, F_n\}$. The co-occurrence feature $C$ is a $n^2$ dimension vector, where each dimension $c_{i,j}$ is the number of the pixel pairs in $R$ satisfying the co-occurrence pattern $(U, V, F_i, F_j)$

$$\begin{gathered} C(U, V) = [c_{1,1}, c_{1,2}, \ldots, c_{1,n}, \ldots, c_{n,n}], \qquad (5.2) \\ c_{i,j} = Count(a, b)\ satisfying\ (U, V, F_i, F_j)\ in\ R,\ 1 \leq i, j \leq n. \end{gathered}$$

As shown in Fig. 5.2, the two axes correspond to the divided score space $F$, and the co-occurrence feature vector has $8 \times 8 = 64$ dimensions.

Compared to the covariance matrix, the co-occurrence features utilize the co-occurrence histogram to describe the distribution of object characteristics instead of the covariance. Our co-occurrence features are based on single co-occurrence pattern, so the extraction will be relatively fast if we adopt efficient methods to generate $F$. In our case, we utilize the methods inherited from Haar, LBP, and HOG to construct the CoHaar, CoLBP, and CoHOG features respectively.

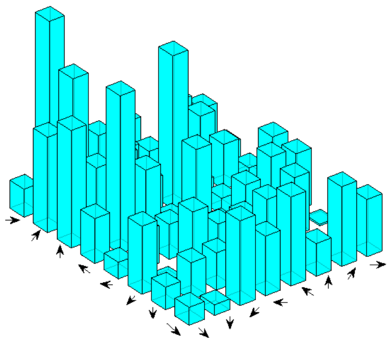


Figure 5.2: $8 \times 8$ co-occurrence histograms.

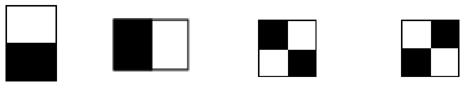


Figure 5.3: Haar features. Black regions have -1 weight, and white regions have +1 weight.

## 5.2 CoHaar feature

Haar-like features [114], shown in Fig. 5.3, consist of two or more rectangular regions enclosed in a template. Such features produce a feature value as

$$F = \sum_{t=1}^{l} w_t I_t, \tag{5.3}$$

where $t$ iterates through all $l$ rectangles, the $I_t$ represents the mean intensity of the pixels enclosed within the $t$th rectangle. Every rectangle in the Haar feature is assigned a weight that is represented by $w_t$. The weights are set such that $\sum_{t=1}^{l} w_t = 0$ is satisfied. The computation of Haar feature is quite efficient because the intensity sum in any rectangles can be easily calculated by the integral image [114].

To construct the CoHaar feature, we extend the Haar feature extraction to the gradient domain. In consideration of the efficiency, we utilize the x and y directional gradient image respectively. The $F$ of CoHaar feature in Eq. 5.2 is replaced with the Haar feature extraction equation 5.3 on the intensity domain, gradient-x domain or gradient-y domain. We quantize $F$ to $n = 8$ bins, so the CoHaar feature dimension is $8 \times 8 = 64$. Given an input window $R$ and an indicator $k$, the CoHaar feature is formulated as

$$\begin{gathered} CoHaar(U, V, k) = [c_{1,1}, c_{1,2}, \ldots, c_{1,8}, \ldots, c_{8,8}] \\ c_{i,j} = Count(a, b)\ satisfying\ (U, V, F_i, F_j)\ in\ R \\ F = \sum_{t=1}^{l} w_t I_t(k), 1 \leq i, j \leq 8, \end{gathered} \tag{5.4}$$

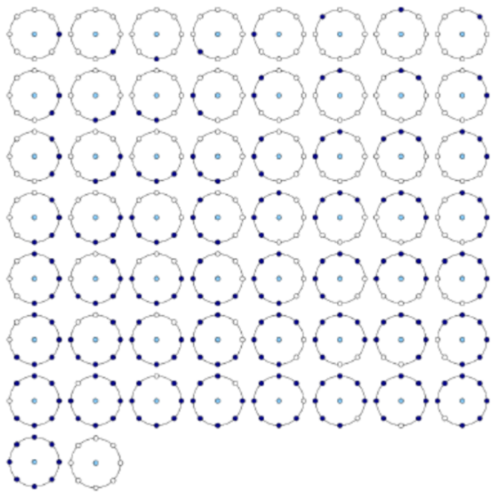


Figure 5.4: 58 uniform patterns of $LBP_{8,1}$ feature. Each row corresponds to a cluster in the CoLBP extraction.

where $F_i, F_j$ are the quantized Haar feature response, $k$ ranges from 0 to 2, $I(k)$ is the intensity sum when $k = 0$, the gradient sum on gradient-x image when $k = 1$, and on gradient-y image when $k = 2$.

## 5.3 CoLBP feature

We have already introduced the non-uniform LBP feature in Chapter 3.1.2. Uniform LBP is a subset of LBP, defined by $\varDelta$ in Eq. 5.5, which shows the number of bitwise transitions from 0 to 1 or vice versa when the bit pattern is considered circular

$$\varDelta(LBP_{d,r}) = |sign(I_{d-1} - I_e) - sign(I_0 - I_e)| + \sum_{i=1}^{d-1} |sign(I_i - I_e|) - sign(I_{i-1} - I_e). \tag{5.5}$$

Fig. 5.4 shows all uniform patterns for $LBP_{8,1}$. The binary patterns are reduced to 59, where all the non-uniform patterns are merged into another pattern.

Ojala et al. [71] has shown that over 90% local structures belong to uniform patterns when using the parameter of $d = 8, r = 1$. Thus, the value calculated by uniform LBP is more stable and less prone to noise. Then we propose the CoLBP based on these uniform patterns. Similarly, the LBP extraction is applied on both the intensity and gradient domain. In consideration of the rotation invariance, we merge the 58 uniform $LBP_{8,1}$ patterns to 8 clusters based on the number of '1' values, shown as the 8 rows in Fig. 5.4. All the non-uniform patterns construct another cluster. As a result, the LBP response space is divided into $n = 9$ bins, so the CoLBP histogram consists of $9 \times 9 = 81$ dimensions. Given an input window $R$ and an indicator $k$, the CoLBP feature vector is generated by

$$\begin{aligned} CoLBP(U,V,k) &= [c_{1,1}, c_{1,2}, \ldots, c_{1,9}, \ldots, c_{9,9}] \\ c_{i,j} &= Count(a,b)\ satisfying\ (U,V,F_i,F_j)\ in\ R \\ F &= LBP_{d,r,k}, 1 \le i,j \le 9, \end{aligned} \tag{5.6}$$

where $F_i, F_j$ are the cluster number of LBP response $F$, $LBP_{d,r,k}$ is the LBP response on intensity image when $k = 0$, on gradient-x image when $k = 1$, and on gradient-y image when $k = 2$.

## 5.4 CoHOG feature

HOG breaks the image region into a cell-block structure and generates histogram based on the gradient orientation and spatial location. Watanabe et al. [119] proposed a dense version extracting all possible co-occurrence patterns of the gradient orientation in the whole image, which is rather time consuming. Instead, we build our CoHOG based on single co-occurrence pattern. The gradient orientation on both the intensity domain and the gradient domain are utilized as the $F$ in CoHOG feature and further quantized to 8 bins. Therefore, there are $8 \times 8 = 64$ elements in the co-occurrence histogram. Given an input window $R$ and an indicator $k$, the CoHOG histogram is formulated as

$$\begin{aligned} CoHOG(U,V,k) &= [c_{1,1}, c_{1,2}, \ldots, c_{1,8}, \ldots, c_{8,8}] \\ c_{i,j} &= Count(a,b)\ satisfying\ (U,V,F_i,F_j)\ in\ R \\ F &= GradientOrientation_k, 1 \le i,j \le 8, \end{aligned} \tag{5.7}$$

where $F_i, F_j$ are the quantized gradient orientation, $F$ is the gradient orientation on original image when $k = 0$, the gradient orientation on gradient-x image when $k = 1$, and on gradient-y image when $k = 2$.

## 5.5 Experiments

We evaluate the proposed method on pedestrian detection and general object detection tasks. For pedestrian detection, the INRIA dataset and Caltech dataset are utilized. The Caltech dataset [23] consists of about 250,000 frames with a total of 350,000 bounding boxes and 2,300 unique pedestrians are annotated. The individuals in these datasets appear in many positions, orientations, and background variety. We use $64 \times 128$ pedestrians and co-

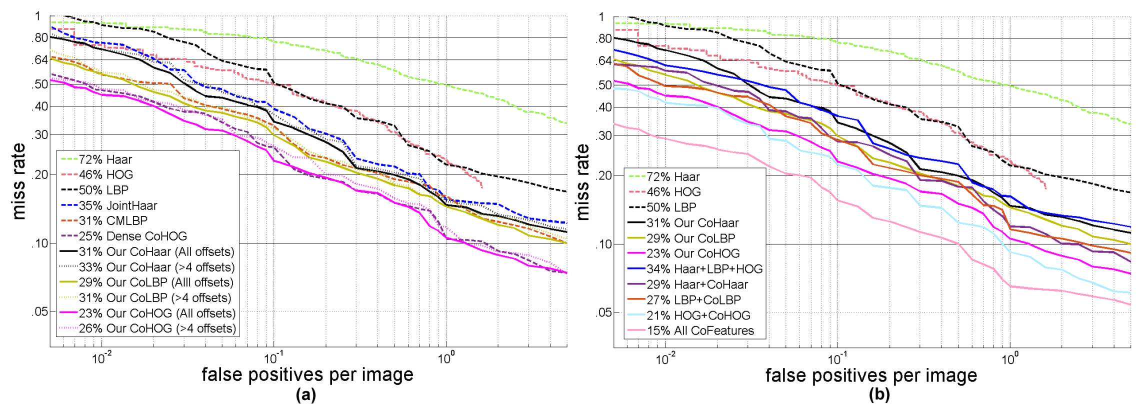


Figure 5.5: Comparison of different co-occurrence features and feature combinations on INRIA pedestrian dataset.

occurrence windows from $6 \times 6$ to $56 \times 112$. The locations of the window centers are sampled every 4 pixels. As a result, this will generate 7, 997 different windows. The evaluation is based on the detection rate versus False Positive rate Per Image (FPPI) [117].

We first evaluate the performance of different co-occurrence features with conventional RealAdaBoost on INRIA dataset. Fig. 5.5(a) illustrates the performance of the boosted classifiers with different co-occurrence features and traditional features. It can be seen that all the detectors with co-occurrence features clearly outperform the detectors with corresponding traditional features. The CoHOG detector performs better than CoLBP and CoHaar detector, which shows that gradient orientation co-occurrence is more discriminative compared to intensity and gradient magnitude co-occurrence. In addition, we notice that the accuracy of the detectors trained on larger offset ($> 4$) CoFeatures are lower, which explains the influence of structure diversity. In fact, 90% offsets of the selected CoFeatures in "CoX (All offsets)" curves are within (4,4). Compared to the existing co-occurrence features, our co-occurrence features are histograms of quantized feature response on selected co-occurrence patterns, which are more discriminative than the combination of Haar responses (JointHaar [66]). In addition, using dense feature vector (CMLBP [70], dense CoHOG [119]) might lead to the dimension redundant. So the accuracy of our co-occurrence features are better.

Next, we compare the combination of the proposed 3 co-occurrence features with other feature combinations. From Fig. 5.5(b) we could find that the combination of the co-occurrence features clearly show better accuracy compared to the combination of low-level features "Haar+LBP+HOG". The groups of "X+Co-X" achieve slightly better compared to using "Co-X", which is still lower than the combination of CoFeatures. The best one among these groups, which combines all co-occurrence feature together, achieves 15% average miss rate. It is a significant improvement compared to using single co-occurrence feature. These results reflect the advantage of using multiple co-occurrence features in pedestrian detection.

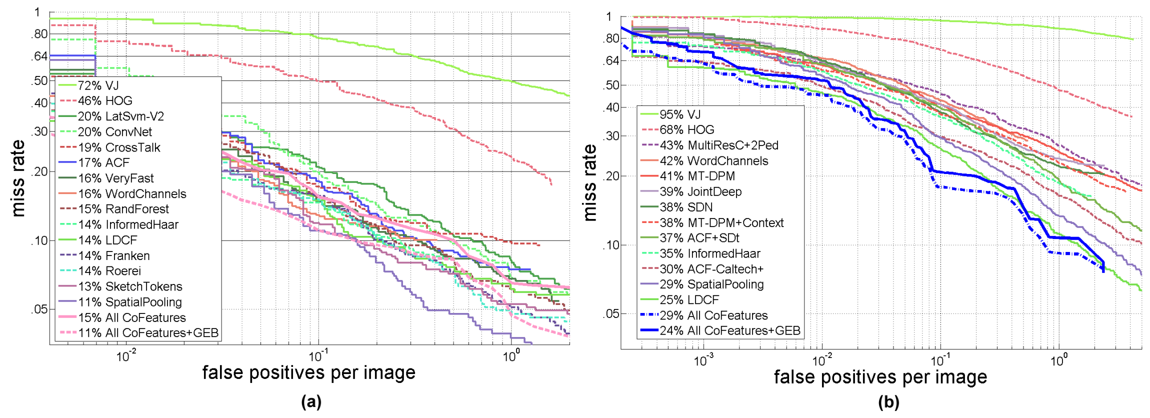


Figure 5.6: Comparison of the co-occurrence features with the commonly-used algorithms. (a) INRIA. (b) Caltech.

Furthermore, we compare our results with the commonly-used methods on pedestrian detection in Fig 5.6. We notice that using the combination of all 3 co-occurrence features, the accuracy is much better than some boosting family methods (Multiftr, FPDW, pAUC-Boost, FisherBoost, Crosstalk), but the detectors with single co-occurrence feature achieve lower accuracy compared to other methods with high accuracy (SketchTokens, Spatialpooling, LDCF, ACF-Caltech+). The reason is that the proposed co-occurrence features are extracted on single scale, so that the discriminative ability might be lower compared to the evolution of channel features, such as the dense sampled multi-scale feature in ACF-Caltech+ [68], or the decorrelated features in LDCF [68]. But this gap can be compensated by the combination of multiple co-occurrence features. In addition, our detector is also better than the MOCO [11] (46% on Caltech) , which is a combination of zero-order, first-order, and second-order co-occurrence information. The major advantage of our method is that we select the meaningful co-occurrence patterns by RealAdaBoost, while MOCO uses all possible patterns with Latent SVM. There will be some redundant information in the dense feature vector.

## 5.6 Conclusion

In this chapter, we introduce how to use the co-occurrence features for better object detection. The proposed co-occurrence features could be used on any traditional low-level features, so that they are quite flexible for general object detection. We show that the accuracy of co-occurrence feature is much better compared to low-level features. It is also comparable with some feature fusion framework. This demonstrates the effectiveness of extracting co-occurrence information on low-level features.

# Chapter 6

# Enhancing the weak classifiers

Boosting integrates several weak classifiers into a strong classifier. The efficiency and accuracy of a boosted classifier is strongly related to the weak classifier. However, since the boosting procedure focuses on the hard samples gradually, it gets more and more difficult to find the weak classifiers that can efficiently improve the classification power of the strong classifier. For those more complicated objects such as the multi-view and multi-pose pedestrian, the problem becomes much more serious that in later training rounds the current classification task might be beyond the ability of the weak classifier [122]. As a result, the training may converge very slowly or can not converge at all.

To solve this problem, one part of our research lays on the weak classifier enhancement, which results in the basis mapping method published in CVPR 2015 [80].

## 6.1 Basis mapping

In the boosting training, the weight of the correctly classified sample is decreased, and the weight of the mis-classified sample is increased. As a result, the weak classifier learning will gradually focus on the hard samples in the training set. It will be more and more difficult to find the appropriate weak classifiers in the later round. Therefore, the effectiveness of the boosting will be improved if we could facilitate the weak classifier training on these hard samples. In this chapter, we plan to restrict the learning of the weak classifier into a constrained region instead of the whole sample space. That means, we propose to map the original samples into the constraint region subject to the hard samples. Intuitively, such mapping "moves" the original samples into a region "around" the hard samples. Therefore, the weak classifier learning can specifically learn the classification hyperplane within the region. This learning task will be much easier compared to that in the whole sample space.

The basis mapping is defined as a process that condenses the sample space into a region referring to a hard sample, namely *basis sample*. It could be formulated as

$$\Phi(\mathbf{x}) = \varphi(\mathbf{x}, \mathbf{x}_b) \quad \varphi : \mathbf{R}^m \times \mathbf{R}^m \to \mathbf{R}^n, \tag{6.1}$$

where $\mathbf{x} \in \mathbf{R}^m$ is the sample in the original space, and $\mathbf{x}_b \in \mathbf{R}^m$ is the basis sample. This $\Phi$ maps the original space $\mathbf{R}^m$ to a new space $\mathbf{R}^n$.

We present a kind of mapping that restricts the mapped samples to a "hypersphere" around the $\Phi(\mathbf{x}_b)$ with the same dimension ($m = n$), and we set the radius as $2||\Phi(\mathbf{x}_b)||$. This mapping could be denoted as

$$\forall \mathbf{x} \in \mathbf{R}^m : ||\Phi(\mathbf{x}) - \Phi(\mathbf{x}_b)|| \leq 2||\Phi(\mathbf{x}_b)||, \tag{6.2}$$

where $\mathbf{x}^{(i)}$ is the $i$th dimension of $\mathbf{x}$, and $||\mathbf{x}||$ represents the sum of all dimensions $\mathbf{x}^{(i)}$ in $\mathbf{x}$

$$\forall \mathbf{x} \in \mathbf{R}^m : ||\mathbf{x}|| \equiv \sum_{i=1}^{m} ||\mathbf{x}^{(i)}||. \tag{6.3}$$

We further give Eq. 6.4, which is a sufficient condition of Eq. 6.2 to constrain the mapping function

$$||\Phi(\mathbf{x}) - \Phi(\mathbf{x}_b)|| \leq ||\Phi(\mathbf{x})|| + ||\Phi(\mathbf{x}_b)|| \leq ||2\Phi(\mathbf{x}_b)||. \tag{6.4}$$

If the mapping satisfies Eq. 6.4, it will also satisfy Eq. 6.2. Therefore, we use Eq. 6.5 as a constraint of the mapping function

$$\forall \mathbf{x} \in \mathbf{R}^m : ||\Phi(\mathbf{x})|| \leq ||\Phi(\mathbf{x}_b)||. \tag{6.5}$$

Substituting $\varphi$ in Eq. 6.1 into Eq. 6.5, we may find that $||\varphi(\bullet, \bullet)||$ is a kind of similarity measure for vectors in $\mathbf{R}^m$. In object detection, histogram features such as HOG and LBP are commonly-used due to their good description ability. If we utilize histogram features as $\mathbf{x}$, the $||\varphi(\bullet, \bullet)||$ could be constructed by histogram distance metrices. Therefore, we adopt three similarity metrics of histograms, the Histogram Intersection, the Chi-Square Distance, and the Bhattacharyya Distance respectively to conduct the function $||\varphi(\bullet, \bullet)||$.

Histogram intersection [63] is usually used as a similarity metric for histogram-based representations of images. It can be formulated as

$$S_{HI}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{m} min(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}). \tag{6.6}$$

Then we define the Histogram Intersection Mapping (HIM). Each dimension of the mapped vector can be calculated as

$$\Phi^{(i)}(\mathbf{x}) = \varphi_{HIM}^{(i)}(\mathbf{x}, \mathbf{x}_b) = min(\mathbf{x}^{(i)}, \mathbf{x}_b^{(i)}). \tag{6.7}$$

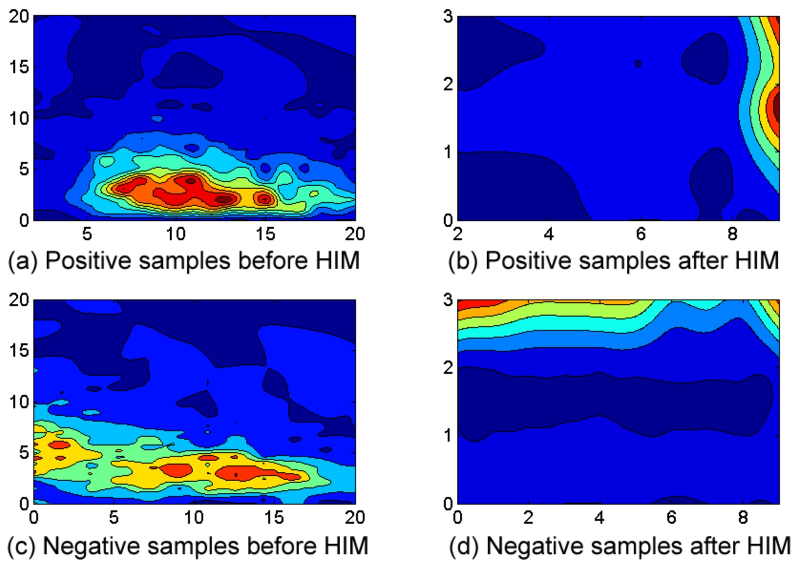


Figure 6.1: Sample distribution before and after Histogram Intersection Mapping (HIM).

Defining the basis mapping based on this measurement has certain advantage because the L-1 distance based measurement is more robust to outliers. To evaluate the effectiveness of the HIM, we train a classifier using HOG feature and LogitBoost algorithm on Caltech pedestrian dataset. The sample distributions on the first selected HOG feature are plotted in Fig. 6.1. The X-axis and the Y-axis represent the two most important dimensions of the HOG feature respectively. They correspond to the two largest weights in the linear regression. Fig. 6.1(a) and 6.1(c) show the positive and negative sample distributions before the HIM. Fig. 6.1(b) and 6.1(d) show the distributions after the HIM referring the basis sample (9, 3) in original space. We may find that it is relatively difficult to use a linear classification plane to separate the positive samples and negative samples in the original space. The HIM maps the original samples into a condensed space, where the pattern distributions become much more separable. It is much easier to learn the classification hyperplane in the mapped space.

The Chi-Square Distance is another distance metric between histograms [3], as formulated in Eq. 6.8

$$S_{CHI}(\mathbf{x},\mathbf{z}) = C - \sum_{i=1}^{m} w^{(m)} \frac{(\mathbf{x}^{(i)} - \mathbf{z}^{(i)})^2}{\mathbf{x}^{(i)} + \mathbf{z}^{(i)}}. \tag{6.8}$$

Then we construct the CHi-square Mapping (CHM) by setting all the $w^{(m)}$ to 1 and omitting the constant $C$

$$\Phi^{(i)}(\mathbf{x}) = \varphi^{(i)}_{CHM}(\mathbf{x},\mathbf{x}_b) = \frac{(\mathbf{x}^{(i)} - \mathbf{x}_b^{(i)})^2}{\mathbf{x}^{(i)} + \mathbf{x}_b^{(i)}}. \tag{6.9}$$

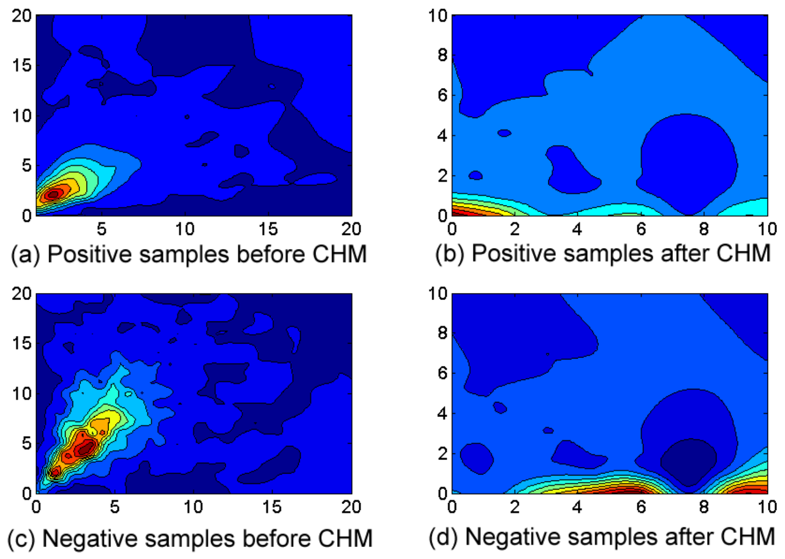


Figure 6.2: Sample distribution before and after CHi-Square Mapping (CHM).

The Bhattacharyya Coefficient [113] can also be employed to measure the similarity between two histograms

$$S_{BHA}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{m} \sqrt{\mathbf{x}^{(i)}\mathbf{z}^{(i)}}. \tag{6.10}$$

Then the BhattaCharyya Mapping (BCM) could be defined as

$$\Phi^{(i)}(\mathbf{x}) = \varphi_{BCM}^{(i)}(\mathbf{x}, \mathbf{x}_b) = \sqrt{\mathbf{x}^{(i)}\mathbf{x}_b^{(i)}}. \tag{6.11}$$

Similarly, the sample distribution of CHM and BCM on Caltech dataset are shown in Fig. 6.2 and Fig. 6.3 respectively, where the basis sample is (2, 3) and (9, 6). It can be seen that both the CHM and BCM map the original samples into a more condense space around the basis sample, which makes the mapped space more appropriate for learning a classification hyperplane. These results show that the CHM and BCM also play the similar role as the HIM.

With these basis mappings, the classification task is moved from original space to a shrinked space around the hard samples. Learning the weak classifiers in the shrinked space will be easier compared to learning in the original space.

## 6.2 Basis Mapping and Kernel Method

Kernel methods map the original samples into an implicit high-dimension space, where the linear classification is subsequently applied. As a result, kernel classifiers achieve better accuracy compared to linear classifiers. In boosted classifier, the overall accuracy is mainly

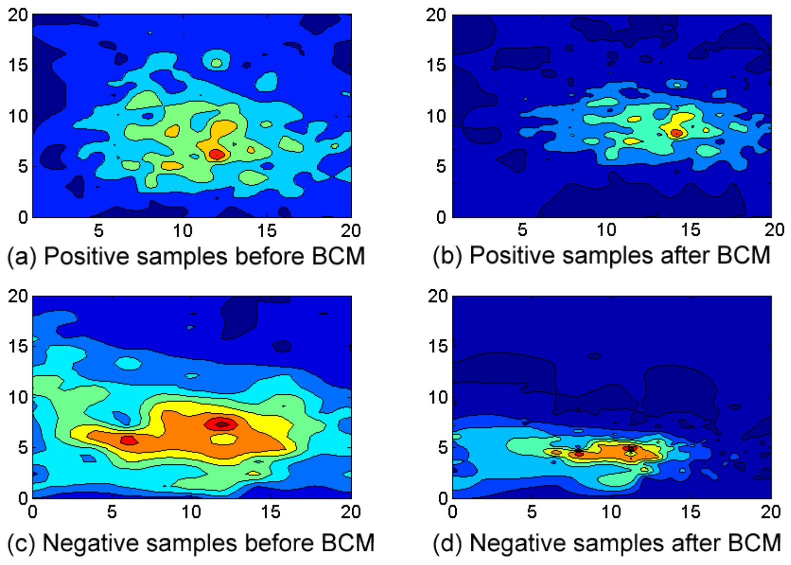


Figure 6.3: Sample distribution before and after BhattaCharyya Mapping (BCM).

decided by the weak classifier [93]. If we utilize the kernel weak classifiers instead of the conventional weak classifiers, the accuracy of the overall boosted classifier will be clearly improved. In this section, we will show that the basis mapping is an approximation of applying additive kernel methods as weak classifiers in the boosting algorithm.

Generally, linear classification in the implicit space can be implemented in the original space through the *kernel trick*. Given two $m$-dimension samples $\mathbf{x}, \mathbf{z}$ in the original space and a kernel function $K(\mathbf{x}, \mathbf{z})$ that satisfies the Mercer's Condition, there exists a function $\psi$,

$$K(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x}) \bullet \psi(\mathbf{z}), \tag{6.12}$$

where $\bullet$ is the dot product of two vectors.

In boosting training, learning weak classifier $h$ could be considered as finding an optimal classification hyper-plane based on the training samples in the original $m$-dimensional space. If the kernel method is applied in the procedure of weak classifier learning, denote the optimal classification hyper-plane in the implicit space by $\mathbf{w}^*$, given a sample in the original $m$-dimension space by $\mathbf{x} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}]$, the optimal weak classifier $h^*(\mathbf{x})$ is the dot product of the $\psi(\mathbf{x})$ and $\mathbf{w}^*$

$$h^*(\mathbf{x}) = \mathbf{w}^* \bullet \psi(\mathbf{x}). \tag{6.13}$$

In the extreme case, if there is a vector $\mathbf{x}^* \in \mathbf{R}^m$ satisfies $\psi(\mathbf{x}^*) = \mathbf{w}^*$, Eq. 6.13 can be implemented by Eq. 6.14

$$h^*(\mathbf{x}) = \mathbf{w}^* \bullet \psi(\mathbf{x}) = \psi(\mathbf{x}^*) \bullet \psi(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}^*). \tag{6.14}$$

Using Eq. 6.14 as the weak classifier is relatively convenient. So the only problem is to find out such an $\mathbf{x}^*$.

Unfortunately, in most of the cases, $\psi$ is not invertible or even $\psi$ itself could not be explicitly described, so it seems to be impossible to find such an $\mathbf{x}^*$. But in boosting framework, we could approximate $\mathbf{x}^*$ by selecting one of the current training samples $\mathbf{x}'$. After evaluating several training samples to select the best one to approximate $\mathbf{x}^*$, the optimal $h^*$ could be approximated as the $h$ in Eq. 6.15

$$h^*(\mathbf{x}) \approx h(\mathbf{x}) = \mathbf{w}' \bullet \psi(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}'), \tag{6.15}$$

where $\mathbf{w}' = \psi(\mathbf{x}')$. This implies that by referring to an appropriate sample $\mathbf{x}'$, the linear classification in the implicit space could be approximated by the above kernel function.

Then we turn back to the basis mapping. In all three proposed basis mappings, each dimension of the feature vector $\mathbf{x}$ is independent with each other, so the basis mapping $\Phi(\mathbf{x})$ could be written as

$$\Phi(\mathbf{x}) = \varphi(\mathbf{x}, \mathbf{x}_b) = [\varphi(\mathbf{x}^{(1)}, \mathbf{x}_b^{(1)}), \ldots, \varphi(\mathbf{x}^{(m)}, \mathbf{x}_b^{(m)})]. \tag{6.16}$$

Notice that all these basis mappings correspond to the additive kernels in Eq. 6.17. The HIM corresponds to the histogram intersection kernel, the CHM corresponds to the chi-square kernel, and the BCM corresponds to the bhattacharyya kernel

$$K(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{m} k(\mathbf{x}^{(i)}, \mathbf{x}'^{(i)}). \tag{6.17}$$

So the $\varphi$ in Eq. 6.16 is exactly the same as the $k$ in Eq. 6.17 for these basis mappings. Then the weak classifier in Eq. 6.15 could be written as

$$h(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{m} \varphi(\mathbf{x}^{(i)}, \mathbf{x}'^{(i)}). \tag{6.18}$$

As mentioned above, in the boosting framework, we could use $\mathbf{x}'$ to approximate $\mathbf{x}^*$. This is achieved by evaluating different hard samples in current training stage to get the best one $\mathbf{x}_b$. Then Eq. 6.18 is achieved by Eq. 6.19

$$h(\mathbf{x}) = \sum_{i=1}^{m} \varphi(\mathbf{x}^{(i)}, \mathbf{x}'^{(i)}) = \sum_{i=1}^{m} \varphi(\mathbf{x}^{(i)}, \mathbf{x}_b^{(i)}). \tag{6.19}$$

We further fit the parameters $a$ and $b$ on Eq. 6.19 as the final weak classifier

```
Parameters
  $N$    number of training samples
  $N_b$    number of basis samples per iteration
  $N_f$    number of features per iteration
  $T$    maximum number of weak classifiers
  $\theta$    threshold of false positive rate
Input: Training set $\{(\mathbf{x}_i, y_i)\}, \mathbf{x}_i \in R^m, y_i \in \{0,1\}$
1. Initialization
   $w_i = 1/N, H(\mathbf{x}_i) = 0, p(\mathbf{x}_i) = 0.5$
2. Repeat for $t = 1, 2, \ldots, T$
  2.1 Compute $z_i$ and $w_i$    2.2 For $m = 1$ to $N_f$
      For $n = 1$ to $N_b$
        2.2.1 Randomly select a basis sample $\mathbf{x}_b$ with top 20% weights
        2.2.2 Calculate the original feature vectors $\mathbf{x}_i$
        2.2.3 Calculate the mapped vectors $\Phi(\mathbf{x}_i) = \varphi(\mathbf{x}_i, \mathbf{x}_b)$
        2.2.4 Fit the function $h$ Eq. 6.20 by weighted least square regression
         from $\Phi(\mathbf{x}_i)$ to $z_i$
        2.2.5 Select the best feature and basis sample with minimum regression error
        2.2.6 Calculate the false positive rate. If it is lower than $\theta$, break
  2.3 Update $H(\mathbf{x}_i)$ and $p(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^{T} h_j(\mathbf{x})]$
```

Figure 6.4: LogitBoost training with basis mapping.

$$h(\mathbf{x}) = \sum_{i=1}^{m} a^{(i)} \varphi(\mathbf{x}^{(i)}, \mathbf{x}_b^{(i)}) + b. \tag{6.20}$$

According to Eq. 6.16, Eq. 6.20 is the linear classification on the mapped space $\Phi(\mathbf{x})$ around the basis sample. The kernel classification Eq. 6.14 is finally transformed to a linear classification. So we get the conclusion that the weak classifier based on the basis mapping $\Phi$ is an approximation of additive kernel classification in the original space, which significantly has better discriminative ability than simple decision stump or linear weak classifiers. Because the performance of a boosted classifier mainly depends on the weak classifiers. The proposed basis mapping will contribute to the overall accuracy of the boosted classifier. In addition, the basis mappings constructed by Eq. 6.6, 6.8, 6.10 are computational efficient, they does not increase the feature dimension. Therefore, the computation cost will not increase much compared to the linear weak classifiers.

## 6.3 LogitBoost based on Basis Mapping

The proposed basis mapping is integrated into the LogitBoost framework as an independent module in each training round. The parameters $a, b$ in the weak classifier $h_j(\mathbf{x})$ are learned by regularized least square. In the basis mapping, the key issue is how to select a hard sample as the basis sample. We know that the hard sample should be the sample close to the classification hyperplane, i.e. $H(\mathbf{x}_i) \approx 0$. It can be seen from Eq. 2.12 that those samples with $p(\mathbf{x}_i) \approx 0.5$ result in $H(\mathbf{x}_i) \approx 0$, which can be considered as the "hard samples". The weight $w_i$ gets the maximum value when $p(\mathbf{x}_i)$ is 0.5, so we consider the samples with top 20% weights as candidate basis samples and randomly select one at each time for the basis mapping.

To learn the best weak classifier, several basis samples and features are evaluated in each iteration. The basis samples will be randomized $N_b = 10$ times. To learn the best feature, the most intuitive way is to look through the whole feature pool, which is rather time consuming. So we resort to a sampling method to speed up the feature selection process. More specifically, as mentioned in [140], a random sub-sample of size $N_f = log0.05/log0.95 = 59$ will guarantee that we can find the best 5% features with a probability of 95%.

The stopping criteria are two folds, either the weak classifier number achieves the maximum number, or the current false positive rate is lower than $10^{-6}$. The current false positive rate could be calculated by the bootstrap. The first bootstrap will be called when 50% of the negative samples are filtered by current strong classifier. Then new samples are bootstrapped to replace the filtered negative samples, and the training is ongoing. Every time 50% of the negative samples are filtered, the bootstrap will be called. This procedure is repeated until one of the two stopping criterions above is satisfied. Fig. 6.4 illustrates the detailed algorithm.

## 6.4 Experiments

### 6.4.1 Histogram features

We utilize the commonly-used histogram features, HOG and LBP histogram to train the boosted classifiers based on basis mapping. For HOG feature, we extract 4 cells ($2 \times 2$) and 8 gradient orientation bins for each candidate block. The final HOG feature dimension is 32. For LBP feature, we extract the uniform patterns of $LBP_{8,1}$, and then generate a 59-dimensional histogram. This histogram consists of 58 uniform patterns of $LBP_{8,1}$, and another pattern which integrates all the non-uniform patterns together. So the LBP feature dimension is 59.

### 6.4.2 Databases

We evaluate the proposed method on pedestrian detection and general object detection tasks. For pedestrian detection, the INRIA dataset and Caltech dataset are utilized. The $64 \times 128$ pedestrians are utilized for the pedestrian detection experiments. For the HOG feature, the cell size ranges from $4 \times 4$ to $28 \times 60$. Both the cell size and the locations of the window centers are sampled every 4 pixels. As a result, 5,672 HOG features are generated for boosting training. For the LBP feature, the window size ranges from $8 \times 8$ to $56 \times 120$, and the window center is also sampled every 4 pixels. So the LBP feature number is the same as HOG. Both of the FPPW ande the FPPI protocols are used for evaluation.

For general object detection, the standard benchmark datasets PASCAL VOC 2007 and 2010, are employed. These two datasets contain images from 20 different categories. The training sample size and feature window size are different for different object categories. For the aeroplane, bird, bottle, chair, diningtable, person, pottedplant, sofa, and TV monitor category, all the samples are used together to train a single detector. For other categories, the training samples are divided into the front/rear view samples and side-view samples according to the aspect ratio, and then train two detectors respectively. The final detection result is based on the NMS of these two detectors. The sample size $(w, h)$ used to train these classifiers are listed in Table 6.1 of Chapter 6.4.5. The size of both the HOG and LBP windows ranges from $8 \times 8$ to $w' \times h'$, where $w', h'$ are the maximum multiple of 4 smaller than $w - 8$ and $h - 8$. As a result, the number of the windows ranges from 1,764 to 5,672. The detection performance is measured using the average precision (AP). A detection result is considered as correct if it has an intersection-over-union ratio of at least 50% with a ground-truth object instance.

### 6.4.3 Experiments on INRIA dataset

Firstly, we use the INRIA dataset to evaluate the performance of different basis mappings. The training samples and testing samples are the same as [19]. In Fig. 6.5, the performance of the boosted classifiers with different basis mappings and without the basis mappings are compared using FPPW protocol. It can be seen that all the algorithms with basis mappings clearly outperform the algorithms without basis mapping for both the HOG, LBP, and the concatenation of these two features. The basis mappings consistently achieve about 3% less miss rates at all FPPW. The accuracy is similar to the curve "X+HIKSVM", which directly using conventional HIKSVM as weak classifier. These results show that the weak classifier based on basis mapping is a good approximation of kernel weak classifier. In addition, the accuracy of HIM and CHM is a bit better compared to BCM, which implies that histogram intersection kernel and chi-square kernel is better compared to bhattacharyya kernel in pedestrian detection. Since conventional boosted HOG works better than boosted LBP, the boosted HOG enhanced by basis mapping is also better compared to the boosted LBP. If we concatenate these two features together, the accuracy could be further improved.

Next, we test the boosted classifiers using different basis samples. The experimental results are illustrated in Fig. 6.5(b). The curves with the "hard" label are trained using the hard samples as the basis samples. The curves with the "random" label are trained with random selecting current training samples as basis samples. It could be seen that the basis mappings with hard samples consistently achieve better accuracy than randomly picking basis samples. So it shows the effectiveness of concentrating the classification around the hard samples.

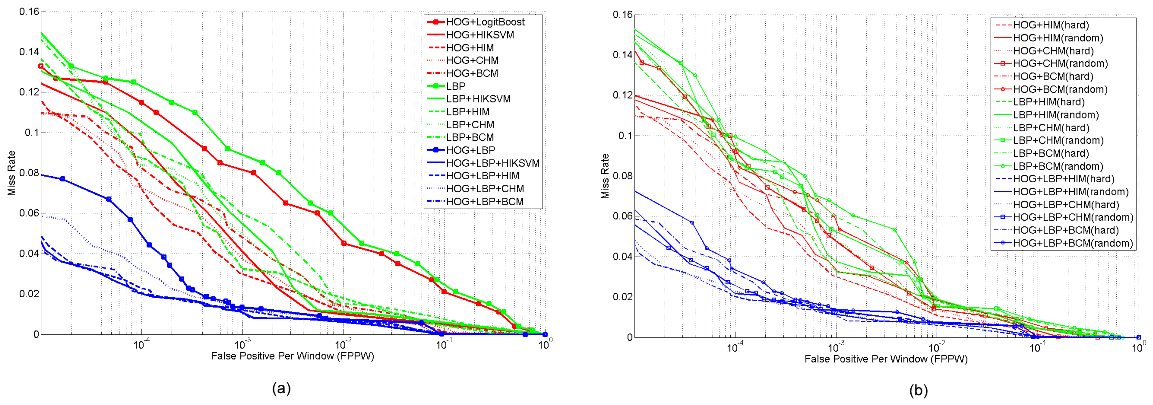


Figure 6.5: FPPW evaluation on INRIA dataset. (a) Comparison of the boosted classifiers based on different basis mappings. (b) Comparison of the boosted classifiers based on different basis samples.

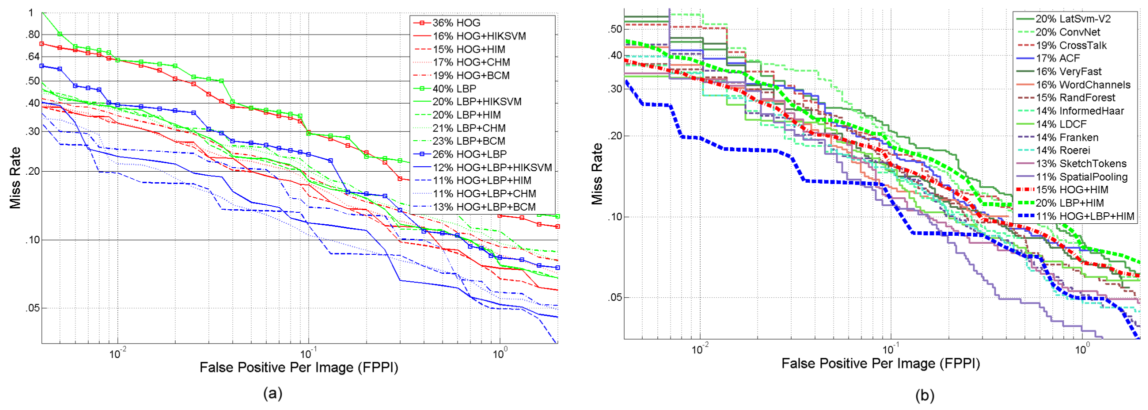


Figure 6.6: FPPI evaluation on INRIA dataset. (a) Comparison of the boosted classifiers based on different basis mappings. (b) Comparison to other commonly-used methods.

Moreover, the above boosted classifiers are evaluated under the criteria of FPPI. In Fig. 6.6(a), it could be seen that the basis mappings clearly improve the detection accuracy of the object detector using the same histogram features and traditional boosting algorithm. The combination of the HOG and LBP shows better accuracy compared to using them individually. The best one, HOG+LBP+HIM, reduces the miss rate from 26% to 11%, which achieves a similar level with using HIKSVM as weak classifier. But the efficiency is much better for both the training and the testing procedure. In Fig. 6.6(b), we compare our methods with the commonly-used methods. Since HIM performs the best in the 3 proposed basis mappings, we only illustrate the HIM curves with different histogram features. It could be seen even using simple HOG or LBP feature, the HIM based boosted classifier achieves 15% and 20% miss rate, which is similar to some methods with complicate features (ConvNet, WordChannels). Our methods also achieve the best accuracy among the boosting family algorithms (Crosstalk, ACF, Veryfast, Wordchannels), which implies that using boosted kernel weak classifiers with simple features is effective. Our result is comparable with the SpatialPooling.

### 6.4.4 Experiments on Caltech dataset

Next, we evaluate our method using the Caltech pedestrian dataset. We follow the training and evaluation protocol proposed by Dollar et al. [23]. The pedestrians at least 50 pixels tall under no or partial occlusion are evaluated. Fig. 6.7(a) illustrates the experimental results of our approach. It could be seen that the accuracy of all the histogram features are significantly improved by the basis mappings. The miss rate of HOG is greatly reduced from 46.9% to 26% (HOG+HIM), 28% (HOG+CHM), and 30% (HOG+BCM). It is similar to the LogitBoost with HIKSVM as weak classifier (27%). Using LBP feature, the miss rate is also reduced about 15%. If we combined the HOG and LBP together, we may achieve 36% miss rate, which is better compared to using HOG and LBP individually. It could be further improved to 22% based on the basis mappings. Among the 3 basis mappings, HIM and CHM still perform better than BCM, which is similar to the experimental results in INRIA dataset.

In Fig. 6.7(b), we compare our methods with the commonly-used algorithms. Using the same HOG feature, our method performs better than MT-DPM. At the same accuracy level (30%-40%), our method is much simpler compared to the methods using complicated features (WordChannels, ACF+SDT). The group with the best accuracy, which integrates HOG and LBP together with HIM, achieves 22% miss rate, which outperforms most of the algorithms listed in this figure. Therefore, we prove the effectiveness of the proposed basis mapping.

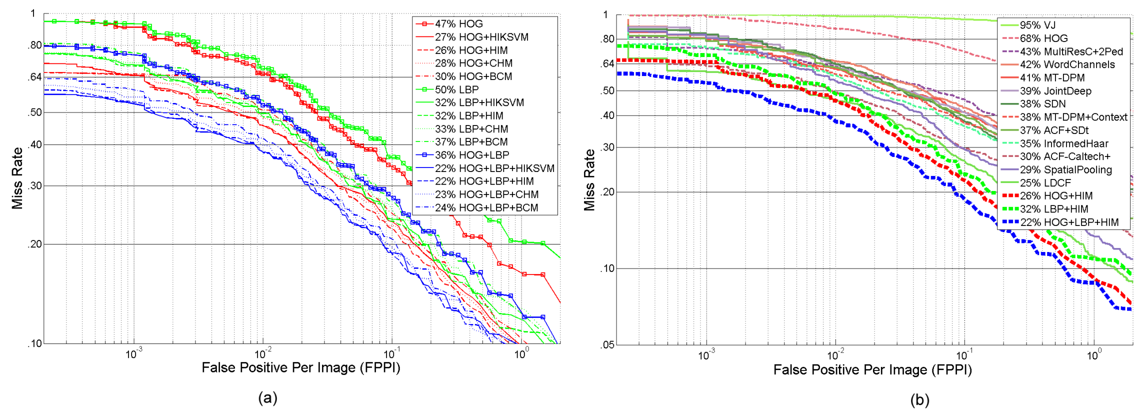


Figure 6.7: FPPI evaluation on Caltech pedestrian dataset. (a) Comparison of the boosted classifiers based on different basis mappings. (b) Comparison to other commonly-used methods.

### 6.4.5 Experiment on PASCAL VOC 2007 and 2010 dataset

Next, we utilize the PASCAL VOC 2007 and 2010 datasets to test our method on general object detection. We follow the requirement of the training set and testing set of the PASCAL challenge [27]. Table 6.1 lists the training sample sizes of the 20 categories. These size are utilized in both the VOC 2007 and VOC 2010 experiments. In Table 6.2 and 6.3, we compare the boosted classifiers based on different basis mappings and histogram features on VOC 2007 and VOC 2010 respectively. From the results we could find that the basis mapping significantly improves the mAP 12% for HOG feature, 8% for LBP feature, and 11% for the combination of these two features. The accuracy is similar to applying kernel SVM as weak classifier, but the speed is much faster. These results are reasonable because it is difficult to solve the general object detection problem by linear classifiers. Using kernel classifier clearly contributes to the overall accuracy. In addition, we find that if the difficulty of the detection task is beyond the description ability of histogram feature, the basis mappings will fail to locate the object. In the experiments, we notice that some of the false positives of the basis mappings are exact the same with the version without basis mapping. This is similar to [115], where the false positives are due to the insufficient description ability of features rather than the classifiers. In addition, we notice that although BCM did not work as well as HIM and CHM in pedestrian detection, the mAP in general object detection is pretty good. It achieves similar mAP compared to HIM on both VOC 2007 and VOC 2010. The best group among the basis mappings, HOG+LBP+HIM, achieves 49.1% mAP on VOC 2007 and 43.6% mAP on VOC 2010, which is greatly improved compared to the detectors using single features. Compared to the top non-deep learning algorithms [11][17][28][118][13] using conventional learning algorithms, it could be seen that even using single HOG feature, our performance is better compare to the SIFT fisher vectors [17], pyramid HOG [28], heterogeneous features [118] and co-occurrence features [11]. Although

Table 6.1: Training sample size of PASCAL VOC 2007 and 2010 datasets.

| | Side-view | Frontal/rear view |
|---|---|---|
| aeroplane | $128 \times 64$ | - |
| bicycle | $128 \times 64$ | $40 \times 100$ |
| bird | $100 \times 100$ | - |
| boat | $100 \times 100$ | - |
| bottle | $40 \times 120$ | - |
| bus | $128 \times 64$ | $100 \times 100$ |
| car | $100 \times 60$ | $100 \times 100$ |
| cat | $100 \times 60$ | $100 \times 100$ |
| chair | $100 \times 100$ | - |
| cow | $100 \times 60$ | 60x100 |
| table | $100 \times 60$ | - |
| dog | $100 \times 60$ | $80 \times 100$ |
| horse | $100 \times 100$ | $60 \times 100$ |
| motor | $128 \times 64$ | $40 \times 100$ |
| person | $60 \times 100$ | - |
| plant | $60 \times 100$ | - |
| sheep | $100 \times 100$ | $60 \times 100$ |
| sofa | $100 \times 100$ | - |
| train | $128 \times 64$ | $100 \times 100$ |
| tv | $100 \times 100$ | - |

the features used in these algorithms are stronger, this blank could be compensated by the proposed basis mapping which enhances the classification to the kernel level. With the combination of HOG and LBP, the accuracy of our methods outperform all other methods listed in these two tables. Compared to the method using same HOG+LBP [13] but different learning method, our accuracy is still 5% better. Compared to the best result in the VOC 2010 challenge [27], the accuracy is also 11% better.

Table 6.2: Experimental results of the basis mappings on PASCAL VOC 2007 dataset.

| **VOC2007 test** | aero | bike | bird | boat | bot. | bus | car | cat | cha. | cow | tab. | dog | hor. | mot. | per. | pla. | she. | sofa | tra. | tv | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HOG | 37.8 | 52.3 | 19.9 | 17.2 | 18.3 | 31.0 | 47.9 | 23.5 | 18.3 | 23.2 | 22.1 | 19.2 | 39.0 | 46.0 | 30.1 | 17.2 | 23.4 | 24.4 | 42.1 | 38.3 | 29.6 |
| HOG+HIKSVM | 54.9 | 58.3 | 26.2 | 25.5 | 31.5 | 56.4 | 62.1 | 42.2 | 22.9 | 35.5 | 47.3 | 30.1 | 55.4 | 54.4 | 46.2 | 20.9 | 37.1 | 41.8 | 50.1 | 51.1 | 42.4 |
| HOG+HIM | 55.5 | 61.9 | 25.6 | 25.3 | 31.4 | 57.2 | 62.4 | 42.9 | 23.7 | 35.2 | 45.1 | 29.6 | 58.4 | 55.6 | 45.1 | 20.9 | 36.8 | 42.5 | 49.0 | 52.8 | 42.8 |
| HOG+CHM | 53.7 | 54.4 | 22.9 | 26.9 | 27.2 | 56.6 | 61.2 | 43.8 | 21.4 | 34.7 | 49.1 | 29.3 | 56.1 | 50.6 | 47.2 | 20.0 | 37.1 | 41.9 | 49.0 | 49.1 | 41.6 |
| HOG+BCM | 54.3 | 60.4 | 26.0 | 24.8 | 30.6 | 56.1 | 61.8 | 44.1 | 25.0 | 36.4 | 50.5 | 28.4 | 59.4 | 55.1 | 43.0 | 19.3 | 37.4 | 40.6 | 49.3 | 51.6 | 42.8 |
| LBP | 28.8 | 43.8 | 20.4 | 18.3 | 19.2 | 31.5 | 49.4 | 24.5 | 19.1 | 24.4 | 22.1 | 21.2 | 38.0 | 37.0 | 33.1 | 18.1 | 24.5 | 20.0 | 43.5 | 39.3 | 28.8 |
| LBP+HIKSVM | 35.5 | 52.9 | 27.2 | 25.1 | 27.7 | 39.2 | 59.1 | 34.0 | 27.9 | 30.2 | 30.1 | 29.6 | 46.4 | 45.3 | 44.1 | 27.3 | 31.8 | 27.5 | 53.0 | 47.8 | 37.1 |
| LBP+HIM | 36.5 | 52.7 | 28.8 | 26.3 | 26.5 | 40.2 | 59.7 | 35.1 | 28.7 | 30.1 | 30.7 | 27.5 | 45.0 | 45.3 | 44.2 | 26.4 | 30.1 | 27.3 | 54.2 | 46.0 | 37.1 |
| LBP+CHM | 35.7 | 50.4 | 26.9 | 26.3 | 27.2 | 41.2 | 58.2 | 33.8 | 26.4 | 30.2 | 30.2 | 29.7 | 45.5 | 43.6 | 45.9 | 25.3 | 32.1 | 28.9 | 53.0 | 47.3 | 36.9 |
| LBP+BCM | 35.0 | 52.0 | 28.1 | 26.5 | 27.0 | 40.4 | 57.2 | 33.8 | 29.2 | 30.5 | 30.1 | 28.9 | 47.7 | 44.8 | 42.1 | 27.3 | 31.1 | 28.4 | 54.1 | 46.3 | 37.0 |
| HOG+LBP | 51.8 | 57.5 | 28.3 | 27.3 | 23.9 | 40.8 | 55.9 | 28.5 | 23.3 | 28.5 | 34.1 | 29.2 | 47.3 | 54.0 | 44.1 | 27.2 | 30.5 | 32.4 | 49.2 | 43.3 | 37.9 |
| HOG+LBP+HIK. | 66.9 | 63.4 | 36.1 | 34.4 | 34.3 | 55.3 | 67.1 | 50.2 | 30.6 | 39.4 | 51.3 | 40.5 | 62.2 | 65.4 | 56.3 | 32.2 | 40.2 | 45.6 | 56.4 | 58.3 | 49.3 |
| HOG+LBP+HIM | 65.3 | 62.9 | 35.9 | 35.0 | 32.4 | 56.2 | 67.4 | 49.9 | 30.7 | 39.2 | 51.1 | 40.0 | 62.4 | 65.6 | 56.5 | 32.3 | 39.8 | 45.5 | 55.9 | 57.7 | 49.1 |
| HOG+LBP+CHM | 62.7 | 64.4 | 36.0 | 32.2 | 34.0 | 54.6 | 66.2 | 49.8 | 30.4 | 38.7 | 51.8 | 39.3 | 61.1 | 64.2 | 56.1 | 31.0 | 40.1 | 45.3 | 55.3 | 55.9 | 48.4 |
| HOG+LBP+BCM | 66.3 | 63.0 | 35.3 | 33.9 | 35.5 | 56.1 | 67.1 | 50.7 | 31.0 | 38.4 | 50.5 | 41.4 | 61.4 | 65.1 | 52.3 | 31.9 | 40.4 | 44.6 | 55.3 | 56.6 | 48.8 |
| Chen [11] | 41.0 | 64.3 | 15.1 | 19.5 | 33.0 | 57.9 | 63.2 | 27.8 | 23.2 | 28.2 | 29.1 | 16.9 | 63.7 | 53.8 | 47.1 | 18.3 | 28.1 | 42.2 | 53.1 | 49.3 | 38.7 |
| Cinbis [17] | 56.1 | 56.4 | 21.8 | 26.8 | 19.9 | 49.5 | 57.9 | 46.2 | 16.4 | 41.4 | 47.1 | 29.2 | 51.3 | 53.6 | 28.6 | 20.3 | 40.5 | 39.6 | 53.5 | 54.3 | 40.5 |
| Wang [118] | 54.2 | 52.0 | 20.3 | 24.0 | 20.1 | 55.5 | 68.7 | 42.6 | 19.2 | 44.2 | 49.1 | 26.6 | 57.0 | 54.5 | 43.4 | 16.4 | 36.6 | 37.7 | 59.4 | 52.3 | 41.7 |
| Felzenszwalb [28] | 36.6 | 62.2 | 12.1 | 17.6 | 28.7 | 54.6 | 60.4 | 25.5 | 21.1 | 25.6 | 26.6 | 14.6 | 60.9 | 50.7 | 44.7 | 14.3 | 21.5 | 38.2 | 49.3 | 43.6 | 35.4 |

Table 6.3: Experimental results of the basis mappings on PASCAL VOC 2010 dataset.

| **VOC2010 test** | aero | bike | bird | boat | bot. | bus | car | cat | cha. | cow | tab. | dog | hor. | mot. | per. | pla. | she. | sofa | tra. | tv | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HOG | 40.6 | 30.9 | 12.1 | 11.7 | 15.7 | 32.3 | 44.6 | 23.1 | 10.1 | 13.1 | 11.7 | 16.5 | 29.3 | 38.3 | 29.4 | 8.3 | 22.9 | 18.1 | 33.6 | 26.2 | 24.9 |
| HOG+HIKSVM | 55.3 | 48.4 | 18.2 | 22.5 | 21.5 | 55.3 | 59.0 | 39.3 | 13.6 | 22.9 | 27.3 | 27.3 | 44.1 | 47.8 | 37.2 | 12.2 | 35.2 | 28.7 | 43.7 | 36.2 | 36.9 |
| HOG+HIM | 55.6 | 48.1 | 17.7 | 22.6 | 21.4 | 54.2 | 59.2 | 38.9 | 13.4 | 23.2 | 25.3 | 26.6 | 44.6 | 47.6 | 35.6 | 11.9 | 34.9 | 27.9 | 43.0 | 36.8 | 36.6 |
| HOG+CHM | 54.3 | 46.4 | 16.5 | 22.0 | 21.2 | 55.0 | 58.9 | 39.8 | 12.0 | 22.7 | 27.4 | 26.5 | 41.7 | 46.2 | 37.1 | 12.0 | 35.0 | 27.3 | 42.6 | 33.1 | 36.0 |
| HOG+BCM | 53.4 | 47.9 | 18.0 | 21.3 | 21.6 | 55.3 | 57.6 | 39.3 | 13.4 | 23.5 | 26.9 | 27.0 | 44.2 | 48.3 | 33.0 | 12.3 | 35.3 | 27.7 | 44.3 | 36.6 | 36.4 |
| LBP | 26.2 | 24.8 | 20.4 | 9.0 | 13.9 | 30.4 | 45.1 | 22.7 | 7.8 | 14.8 | 9.3 | 13.9 | 26.9 | 29.4 | 24.3 | 6.7 | 18.5 | 14.2 | 31.4 | 24.3 | 22.1 |
| LBP+HIKSVM | 34.7 | 37.7 | 13.4 | 16.2 | 18.4 | 38.5 | 54.0 | 32.5 | 11.3 | 20.1 | 17.6 | 21.3 | 36.1 | 35.6 | 34.0 | 10.1 | 28.7 | 21.3 | 39.0 | 32.5 | 29.5 |
| LBP+HIM | 33.9 | 36.5 | 13.1 | 15.7 | 18.1 | 38.9 | 54.6 | 32.1 | 11.7 | 20.1 | 16.9 | 21.3 | 35.0 | 33.2 | 34.2 | 9.4 | 28.1 | 23.6 | 39.2 | 31.9 | 29.2 |
| LBP+CHM | 34.3 | 37.7 | 13.4 | 16.3 | 18.4 | 38.5 | 53.7 | 31.8 | 10.4 | 20.7 | 17.2 | 19.6 | 35.6 | 34.1 | 35.1 | 9.3 | 29.0 | 24.2 | 38.3 | 32.3 | 29.3 |
| LBP+BCM | 34.5 | 37.5 | 13.7 | 16.5 | 17.9 | 39.1 | 54.4 | 32.5 | 11.2 | 20.8 | 17.6 | 21.9 | 36.4 | 35.3 | 31.5 | 9.3 | 28.8 | 25.0 | 39.1 | 31.3 | 29.5 |
| HOG+LBP | 47.4 | 39.2 | 15.5 | 18.0 | 19.6 | 37.9 | 46.5 | 28.1 | 14.0 | 18.7 | 17.4 | 24.5 | 36.8 | 45.1 | 32.3 | 9.9 | 28.4 | 22.4 | 35.8 | 27.9 | 30.2 |
| HOG+LBP+HIK. | 63.3 | 53.9 | 26.1 | 27.4 | 26.8 | 57.2 | 56.2 | 49.2 | 20.4 | 30.4 | 34.2 | 38.3 | 49.9 | 58.5 | 47.3 | 14.2 | 41.0 | 34.7 | 51.0 | 45.3 | 43.7 |
| HOG+LBP+HIM | 61.2 | 53.9 | 26.1 | 26.8 | 25.3 | 57.5 | 56.4 | 49.6 | 20.9 | 31.0 | 34.5 | 38.0 | 49.7 | 58.0 | 47.5 | 14.3 | 40.8 | 34.2 | 51.3 | 45.7 | 43.6 |
| HOG+LBP+CHM | 62.7 | 52.2 | 24.3 | 27.0 | 26.0 | 55.3 | 56.2 | 48.5 | 20.7 | 30.7 | 33.7 | 37.5 | 50.2 | 57.3 | 45.9 | 14.0 | 39.9 | 33.3 | 51.3 | 41.9 | 42.9 |
| HOG+LBP+BCM | 62.5 | 54.3 | 26.7 | 26.1 | 26.7 | 56.4 | 55.9 | 49.0 | 20.2 | 30.6 | 33.9 | 37.9 | 49.6 | 56.9 | 44.9 | 14.9 | 41.3 | 34.6 | 51.7 | 44.6 | 43.4 |
| Chen [13] | 54.6 | 53.7 | 16.2 | 12.5 | 31.2 | 54.0 | 44.2 | 40.0 | 16.7 | 32.2 | 29.1 | 30.1 | 54.3 | 57.2 | 43.9 | 12.5 | 35.4 | 28.8 | 51.1 | 40.7 | 36.9 |
| Everingham [27] | 56.7 | 39.8 | 16.8 | 12.2 | 13.8 | 44.9 | 36.9 | 47.7 | 12.1 | 26.9 | 26.5 | 37.2 | 42.1 | 51.9 | 25.7 | 12.1 | 37.8 | 33.0 | 41.5 | 41.7 | 32.9 |
| Wang [118] | 65.0 | 48.0 | 25.9 | 24.6 | 24.5 | 56.1 | 54.5 | 51.2 | 17.0 | 28.9 | 30.2 | 35.8 | 40.2 | 55.7 | 43.5 | 14.3 | 43.9 | 32.6 | 54.0 | 45.9 | 39.7 |
| Felzenszwalb [28] | 49.2 | 53.8 | 13.1 | 15.3 | 35.5 | 53.4 | 49.7 | 27.0 | 17.2 | 28.8 | 14.7 | 17.8 | 46.4 | 51.2 | 47.7 | 10.8 | 34.2 | 20.7 | 43.8 | 38.3 | 33.4 |

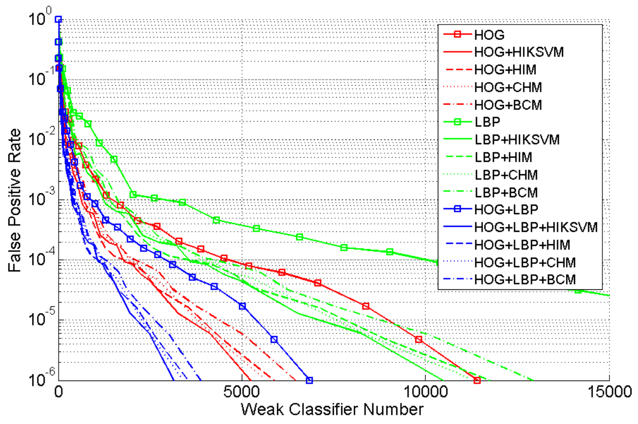


Figure 6.8: Convergence speed of the boosted object detectors based on different basis mappings on INRIA pedestrian dataset.

## 6.4.6 Speed analysis

Next, we analyze the speed issues of the basis mappings. Firstly, Fig. 6.8 plots the false positive rate against the number of weak classifiers for detectors trained on INRIA dataset. All the detectors are trained to $10^{-6}$ false positive rate. This figure shows that it is difficult for the conventional training using LogitBoost with histogram features to converge, especially when the FPPW is lower. On the other hand, the algorithms with the basis mappings converge faster for all histogram features. Among them, the HIM algorithm is found to be the fastest, at the rate of approximately three times faster than the conventional training without any basis mapping. The convergence speed is a little slower than LogitBoost with HIKSVM, which is due to the fact that the basis mapping is an approximation rather than exact equivalence. In general, the performance of boosted classifier is shown to be positively proportional to the convergence speed in training. This signifies that the proposed basis mapping can consistently enhance the training accuracy and the training speed of boosted classifiers. It also somehow explains why the accuracy of BCM is a bit lower compared to HIM and CHM in pedestrian detection.

Table 6.4: Training and testing speed of the boosted classifiers with and without basis mapping.

| **Approach** | **Training speed** | **Patches/sec** |
|---|---|---|
| HOG | 47 hours | 167k |
| HOG+HIKSVM | 242 hours | 8k |
| HOG+HIM | 14 hours | 181k |
| HOG+CHM | 15 hours | 174k |
| HOG+BCM | 18 hours | 154k |
| LBP | 43 hours | 247k |
| LBP+HIKSVM | 211 hours | 10k |
| LBP+HIM | 12 hours | 255k |
| LBP+CHM | 14 hours | 249k |
| LBP+BCM | 16 hours | 234k |
| HOG+LBP | 55 hours | 122k |
| HOG+LBP+HIKSVM | 342 hours | 5k |
| HOG+LBP+HIM | 20 hours | 137k |
| HOG+LBP+CHM | 22 hours | 129k |
| HOG+LBP+BCM | 26 hours | 128k |

We test the training and detection speed of the above INRIA detectors on an Intel I7 dual core PC with 8GB memory. The results are listed in Table 6.4. It could be seen that the training speed is greatly improved using the basis mapping compared to using conventional linear weak classifiers, while the testing speed is still similar. Since there is a root operation in the BCM, the execution time of BCM will be a bit slower compared to HIM and CHM. Compared to the boosting utilizing efficient HIKSVM [63] as weak classifier, both the training speed and testing speed are far better. Compared to the boosting with linear weak classifier, the training speed is 3 times improved. The testing time is also slightly improved because there are few weak classifiers in the resulting detector. In addition, although LBP extraction is faster compared to HOG, the training time is similar because the number of the LBP weak classifier is 2-3 times greater than HOG weak classifiers when converging to $10^{-6}$ false positive rate. So we can get the conclusion that the basis mapping contributes to the accuracy and efficiency at the same time.

## 6.5 Conclusion

In this chapter, we describe how to improve the weak classifier learning in boosting procedure through the proposed basis mapping method. The basis mapping is capable of improving accuracy and training speed of boosted classifiers. The original samples are mapped to a closed, restricted local region defined by the hard-to-classify samples in current stage. Such mapping modifies the distribution of the samples so that the classification performed on mapped space is enhanced over classification on original un-mapped space. This results in more efficient boosting. In addition, we show the relationship of the basis mapping and kernel method. The linear classification in the mapped space achieves similar performance with the non-linear classification in the original space. Three basis mappings (namely HIM, CHM, and BCM) are proposed and shown their effectiveness on INRIA, Caltech, PASCAL VOC 2007 and 2010 datasets. With the commonly-used HOG and LBP histograms, the resulting classifier achieves the best results among the detectors based on conventional machine learning algorithms.

# Chapter 7

# Efficiency-accuracy trade-off

Another part of our research is dealing with the efficiency-accuracy trade-off in boosted training. Most of the traditional machine learning algorithms focus on the accuracy. They always prefer the high discriminative features. In general, such features may solve some complicate object detection tasks, but they will also lead to low generalization ability and poor efficiency. If the generalization ability is low, the detector will fail to process the cases which are different from the training set. If the efficiency is low, the resulting detector could not be applied in real systems. So solving the efficiency-accuracy trade-off problem is important.

## 7.1 Generalization and Efficiency Balanced (GEB) framework

Our first work is the Generalization and Efficiency Balanced (GEB) framework. This work aims to balance the discriminative ability, generalization, and efficiency in the feature selection procedure. It is part of our ICCV submission [85], together with the co-occurrence features in Chapter 5.

### 7.1.1 GEB framework with RealAdaBoost

For the binary object/background classification problem, denote the input data as $(\mathbf{x_i}, y_i), 1 \leq i \leq n$ where $\mathbf{x}_i$ is the training sample and $y_i \in \{-1, 1\}$ is the class label. Each feature can be seen as a function from the image space to a real valued range. We utilize RealAdaBoost to train the boosted classifiers. As mentioned in Eq. 2.11, the feature selection protocol In RealAdaBoost is

$$Z = 2\sum_j \sqrt{W_+^j W_-^j}. \tag{7.1}$$

If the discriminative ability is the only objective, using the features minimizing Eq. 7.1 seems to be a good idea. In our case, the generalization power and computation cost are also considered. Firstly we discuss the influence of the generalization power. The classification margin of the weak classifier $h$ on $\mathbf{x}$ is $y \cdot h(\mathbf{x})$, where the $h$ is normalized to $[-1, 1]$. This margin represents the classification ability of the classifier. Larger margins imply lower generalization error [94]. So we define the term used to evaluate the influence of the generalization power as

$$S(h, \mathbf{x}) = \frac{y \cdot h(\mathbf{x})}{s}, \tag{7.2}$$

where $s$ is a parameter related with the features used to construct this weak classifier. For example, if we use the co-occurrence features, $s$ is related with the offsets $(U, V)$ in co-occurrence pattern, calculated as

$$s = \begin{cases} 1 & max(U,V) \leq \delta \\ 1.5 - \dfrac{1}{1 + \exp^{\delta - max(U,V)}} & max(U,V) > \delta \end{cases}.$$

This equation means that we believe the co-occurrence patterns within $\delta$ pixels offset are confident. Balancing the generalization power and the discriminative ability requires us to evaluate both Eq. 7.1 and Eq. 7.2. So we add a generalization penalty term into the Eq. 7.1, where $\alpha$ is the generalization-aware factor

$$Z = 2\sum_j \sqrt{W_+^j W_-^j} - \frac{\alpha}{n \cdot s} \sum_{i=1}^{n} y \cdot h(\mathbf{x}_i). \tag{7.3}$$

If the confidence of the selected feature is lower, which corresponds to a smaller margin and larger $s$. Then the second term will be smaller, and $Z$ will be larger. So this feature will have less probability to be selected.

Then we discuss the influence of the computation cost. In real object detection, an object detector will go around the input image to check every candidate detection window. The number of the false positive windows is far larger compared to the true positive windows, especially at the beginning stages. As a result, the execution time of the whole detection procedure mainly depends on the number of false positive windows

$$T \approx \sum_{i=1}^{l} N_{neg,i} t_i, \tag{7.4}$$

where $l$ is the stage number, $N_{neg}$ is the number of false positive windows, $t$ is the computation cost of the weak classifiers. Because $N_{neg}$ depends on the current false positive rate, Eq. 7.4 is equal to Eq. 7.5, where $N$ is the total window number, $fp_i$ is the false positive rate of the $i$th stage

$$T \approx \sum_{i=1}^{l} N fp_i t_i = N \sum_{i=1}^{l} fp_i t_i. \tag{7.5}$$

Then we add another term into Eq. 7.3 as

$$Z = 2 \sum_j \sqrt{W_+^j W_-^j} - \frac{\alpha}{n \cdot s} \sum_{i=1}^{n} y \cdot h(\mathbf{x}_i) + \beta \cdot fp \cdot t, \tag{7.6}$$

where $\beta$ is the efficiency-aware factor. The trade-off of discriminative ability and efficiency in Eq. 7.6 can be explained as follows: in the beginning stages of RealAdaBoost, because the false positive rate is larger, and the target object is still easy to be classified with the background, so RealAdaBoost will refer to efficient features. In the following stages when the false positive rate is smaller and the problem becomes more difficult, the features with higher computation cost will be considered. This strategy makes sense, because the overall efficiency of a cascade boosted classifier is mainly influenced by the beginning stages, which filter most of the negative windows. Using efficiency features in the beginning stages clearly contributes to the overall efficiency. Using the GEB framework, we are able to construct the classifier based on the features not only has sufficient discriminative ability, but also extracted efficiently.

### 7.1.2 Integrating co-occurrence features with GEB framework

We train the boosted classifier based on co-occurrence features and GEB framework. The training procedure is illustrated in Fig. 7.1. To learn the best feature, the most intuitive way is to look through the whole feature pool, which is rather time consuming. So we sample $M = 60$ windows per iteration to speed up the feature selection process. The offsets $(U, V)$ range from (1,1) to (15,30). $O = 15$ offsets are sampled per window, while at least 5 of them are within $(4, 4)$. For CoHaar feature, the four patterns illustrated in Fig. 5.3 are utilized, and the block size of single Haar feature ranges from $4 \times 4$ to $20 \times 20$. For CoLBP feature, the block size is set from $1 \times 1$ (traditional LBP) to $8 \times 8$ (LBP variant). After picking a window and an offset, a random co-occurrence feature is generated and evaluated according to Eq. 7.6. We set the computation cost of CoHaar to 2, CoLBP to 4, and CoHOG to 10. The generalization-aware factor $\alpha$ is set to 0.1, and the efficiency-aware factor $\beta$ is set to 0.15, which is decided by the experimental results of several detectors with different parameters trained on Caltech database.

In the training process, the first bootstrap will be called when 50% of the negative samples are filtered by current strong classifier. Then new samples are bootstrapped to replace the filtered negative samples, and the training is ongoing. Every time 50% of the negative samples are filtered, the bootstrap will be called. This procedure is repeated until the overall fp is lower than $10^{-6}$ or the number of weak classifiers exceeds $T$.

```
Parameters
  N    number of training samples
  M    number of evaluated windows each iteration
  O    number of evaluated offsets each iteration
  T    maximum number of weak classifiers
Input: Training set $\{(\mathbf{x}_i, y_i)\}, y_i \in \{-1, 1\}$
1. Initialize sample weight and classifier output
   $w_i = 1/N, H(\mathbf{x}_i) = 0$
2. Repeat for $t = 1, 2, \ldots, T$
  2.1 Update the sample weight $w_i$ using the $t^{th}$ weak
   classifier output $w_i = w_i e^{-y_i h_t(\mathbf{x}_i)}$
  2.2 For $m = 1$ to M
      For $o = 1$ to O
       For $k = 0$ to 2
        2.2.1 Generate a random $R$ and $(U, V)$
        2.2.2 Calculate feature response $C(U, V, k)$ on $R$
        2.2.3 Build the $W_+$ and $W_-$
        2.2.4 Select the best feature minimizing $Z$ in Eq. 7.6
        2.2.5 If the false positive rate is lower than $10^{-6}$, break
  2.3 Update weak classifier $h_t(x)$ using
  2.4 Update strong classifier $H_{t+1}(\mathbf{x}_i)$
3. Output classifier $H(\mathbf{x}_i) = sign[\sum_{j=1}^{t} h_j(\mathbf{x})]$
```

Figure 7.1: Selecting co-features using RealAdaBoost with GEB.

### 7.1.3 Experiments

We apply the experiments on several computer vision tasks to show the effectiveness of GEB, including pedestrian detection, general object detection, gender recognition, and age estimation.

#### Datasets

For pedestrian detection, the Caltech dataset is utilized. We use $64 \times 128$ pedestrians and co-occurrence windows from $6 \times 6$ to $56 \times 112$. The locations of the window centers are sampled every 4 pixels. As a result, this will generate $7,997$ different windows. The evaluation is based on the detection rate versus False Positive rate Per Image (FPPI) [117].

For general object detection, the standard benchmark dataset PASCAL VOC 2007, is employed. Similar to the experiment setting in Chapter 6.4.2, the training samples are divided into the front/rear view samples and side-view samples according to the aspect ratio. The final detection result is based on merging the outputs of the detectors based on different aspect ratio. The sample size $(w, h)$ used to train these detectors are the same as Table 6.1. We also list them in the second column of Table 7.2 for reference.

For gender recognition, FERET and LFW databases and are utilized to show the effectiveness of the feature fusion and the GEB framework. The FERET database [75] contains gray scale images of 1,199 individuals with uniform illumination but different poses, and the LFW database [45] contains 13,233 color face photographs of 5,749 subjects collected from the web. We manually label the groundtruth regarding gender for each face. The faces that are not (near) frontal, with rotation larger than 45 degree, small scale, and strong occlusion are not considered.

For age estimation, the PAL and FG-NET databases are utilized. The PAL aging database [65] contains 430 18-93 years old Caucasians. This database includes various expressions such as smiling, sadness, anger, or neutral faces. We use only neutral faces in order to exclude the facial expression effect. The FG-NET aging database [31] is one of the most frequently used database for estimating age in the previous works. This database has 1,002 images composed of 82 0-69 years old Europeans. Individuals in the database have one or more images included at different ages. There are also extreme variations in lighting, expression, background, pose, resolution and noise.

### Pedestrian detection

We first conduct the experiments on Caltech dataset. Fig. 7.2 gives the results of the conventional detectors, the detectors with the generalization penalty (gen.) and efficiency penalty (eff.) respectively, and using the whole GEB. Firstly we notice that the co-occurrence features also work better compared to traditional features and their combinations on Caltech database. Using the generalization penalty, the accuracy is improved at least 3% for both single co-occurrence feature and its combinations. Using the efficiency penalty term will not influence the accuracy very much. We know that in image-based object detection, the overall accuracy is decided not only by the discriminative ability of the detector, but also by the generalization power. So using the GEB framework to balance them could contribute to the accuracy of the resulting classifier.

Moreover, We test the resulting classifiers on a desktop dual core I7 PC with 8 GB memory. The average execution speed for $640 \times 480$ images is shown in Table 7.1. It could be seen that although CoHOG has better discriminative ability compared to CoHaar and CoLBP, but the computation is relatively slow because there is no efficient implementation on the algorithm level to get the gradient orientation co-occurrence. If we combine these features together and use the GEB framework, the execution time is significantly reduced from 82.2ms per frame to 50.8ms per frame. These results show the effectiveness of the GEB framework on improving the efficiency.

### General object detection

In Table 7.2, we compare our method with the top methods using local features on PASCAL VOC 2007 dataset, in terms of detection AP on the test set. The GEB is applied for all the CoHaar, CoHOG, and CoLBP here. Firstly, it could be seen that the mAP of using

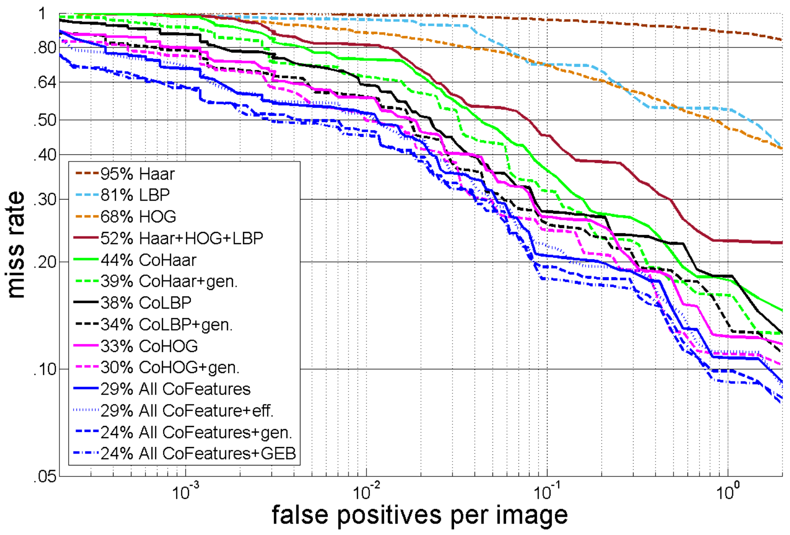


Figure 7.2: Experiments of GEB framework on Caltech dataset.

Table 7.1: Execution speed of the detectors trained by co-occurrence features on Caltech database.

| **Approach** | **Detection time(ms) per frame** |
|---|---|
| CoHaar | 27.5 |
| CoLBP | 36.5 |
| CoHOG | 89.4 |
| All features | 82.2 |
| All features+GEB | 50.8 |

Table 7.2: Average precision(%) of the co-occurrence features + GEB on PASCAL VOC 2007 dataset. In the second column, the 'f' means the sample size for front/rear images.

| | **Sample size** | CoHaar | CoLBP | CoHOG | **All** | **All+GEB** | [11] | [17] | [118] | [28] |
|---|---|---|---|---|---|---|---|---|---|---|
| aero | 128 × 64 | 30.1 | 39.7 | 42.3 | 51.5 | **56.7** | 41.0 | 56.1 | 54.2 | 36.6 |
| bic. | 128 × 64 40 × 100(f) | 45.5 | 51.2 | 59.9 | 59.7 | 61.6 | **64.3** | 56.4 | 52.0 | 62.2 |
| bird | 100 × 100 | 22.3 | **26.3** | 25.4 | 24.8 | 26.0 | 15.1 | 21.8 | 20.3 | 12.1 |
| boat | 100 × 100 | 16.3 | 21.6 | **27.0** | 24.2 | 26.6 | 19.5 | 26.8 | 24.0 | 17.6 |
| bot. | 40 × 120 | 18.9 | 25.9 | 25.1 | 28.1 | 30.9 | **33.0** | 19.9 | 20.1 | 28.7 |
| bus | 128 × 64 100 × 100(f) | 31.3 | 43.9 | 52.2 | 55.9 | **58.2** | 57.9 | 49.5 | 55.5 | 54.6 |
| car | 100 × 60 100 × 100(f) | 46.2 | 58.4 | 55.9 | 62.6 | 66.5 | 63.2 | 57.9 | **68.7** | 60.4 |
| cat | 100 × 60 100 × 100(f) | 27.2 | 38.6 | 38.0 | 41.9 | 44.2 | 43.8 | **46.2** | 42.6 | 25.5 |
| cha. | 100 × 100 | 19.1 | **24.6** | 23.0 | 23.7 | 24.5 | 23.2 | 16.4 | 19.2 | 21.1 |
| cow | 100 × 60 60 × 100(f) | 22.5 | 29.5 | 36.9 | 37.7 | 40.9 | 28.2 | 41.4 | **44.2** | 25.6 |
| din. | 100 × 60 | 26.0 | 33.4 | 41.7 | 43.8 | 44.0 | 29.1 | 47.1 | **49.1** | 26.6 |
| dog | 100 × 60 80 × 100(f) | 18.6 | 21.8 | 24.9 | 26.7 | **29.3** | 16.9 | 29.2 | 26.6 | 14.6 |
| hor. | 100 × 100 60 × 100(f) | 39.1 | 46.8 | 49.7 | 51.2 | 55.3 | **63.7** | 51.3 | 57.0 | 60.9 |
| mot. | 128 × 64 40 × 100(f) | 44.2 | 48.5 | 48.3 | 49.1 | **54.5** | 53.8 | 53.6 | **54.5** | 50.7 |
| per. | 60 × 100 | 29.4 | 44.7 | **48.6** | 47.2 | 48.4 | 47.1 | 28.6 | 43.4 | 44.7 |
| pla. | 60 × 100 | 15.5 | 19.6 | 17.7 | 20.5 | **21.2** | 18.3 | 20.3 | 16.4 | 14.3 |
| she. | 100 × 100 60 × 100(f) | 32.3 | 30.9 | 35.4 | 34.9 | 37.7 | 28.1 | **40.5** | 36.6 | 21.5 |
| sofa | 100 × 100 | 22.9 | 30.9 | 37.9 | 39.7 | **42.7** | 42.2 | 39.6 | 37.7 | 38.2 |
| tra. | 128 × 64 100 × 100(f) | 39.6 | 49.2 | 48.9 | 50.0 | 53.8 | 53.1 | 53.5 | **59.4** | 49.3 |
| tv | 100 × 100 | 28.4 | 38.2 | 42.3 | 43.8 | 47.5 | 49.3 | **54.3** | 52.3 | 43.6 |
| mAP | - | 28.5 | 36.2 | 39.1 | 40.8 | **43.7** | 38.7 | 40.5 | 41.7 | 35.4 |

all CoFeatures is 0.408, which is better compared to using single co-occurrence feature. So combining boosted co-occurrence features is also effective for general object detection. In addition, we notice that the co-occurrence features based on binary information (CoHaar, CoLBP) work relatively well on some object categories with specific structural information, such as the pottedplant with a consistent base, or the chair which consists of several rigid parts. In this case, such co-occurrence features are easier to capture the binary information. In contrast, the gradient information based CoHOG works better on the object categories with complicate appearance such as sheep or sofa. Compared to other methods, the combination of the CoFeatures are more effective than pyramid HOG [28], MOCO [11], SIFT fisher vectors [17], and heterogeneous features [118] including multi-scale HOG and covariance matrix. It implies that using the combination of co-occurrence features is better compared to the combination of traditional features.

### Gender recognition

We utilize the SIFT, HOG, Gabor and LBP feature together with the GEB framework for gender recognition. Since it is difficult to generalize structure diversity term in the feature fusion framework, we remove the generalization term, and only consider the efficiency-accuracy trade-off. The computation cost $C$ for these 4 features are set to 25, 5, 9, and 1 respectively.

For FERET database, similar to Makinen and Raisamo's work [64], the faces of the Fa subset are used and the duplications are eliminated. Therefore, 199 female and 212 male images are used. A 5-fold cross validation testing scheme is applied, where the same ratio between male and female faces are kept for each fold. 15 classifiers are trained, which include the classifiers utilizing single feature (e.g., SIFT), the classifiers using the combination of two features (e.g., HOG+LBP), the classifiers with the combination of 3 features, and the classifiers using all 4 features. The experimental results are shown in the second column of Table 7.3. Firstly, we notice that the accuracy of LBP is lower compared to other features, which is due to the fact that the gradient information is more effective than the binary information in the gender recognition. In addition, it can be found that using multiple features, the average recognition rate is clearly improved compared to using single feature. Both of LBP and Gabor are good complementary to gradient features. This result is reasonable, because the local edge characteristic extracted by the gradient features could be complemented by different information extracted by Gabor or LBP. Moreover, the GEB framework will not influence the overall accuracy much for all the feature combinations, which indicates that it emphasizes the accuracy and the efficiency at the same time. Compared to other methods [106][24][4] which are also based on multiple complicated features, our method achieves competitive accuracy. These results show that the feature fusion with GEB is effective for gender recognition.

Table 7.3: Gender recognition on FERET and LFW database.

| **Approach** | **FERET** | **LFW** |
|---|---|---|
| SIFT | 94.89% | 94.62% |
| HOG | 92.18% | 93.23% |
| Gabor | 93.77% | 94.94% |
| LBP | 82.44% | 81.12% |
| SIFT + HOG | 95.86% | 96.15% |
| SIFT + HOG + GEB | 95.86% | 96.10% |
| SIFT + Gabor | 96.45% | 96.93% |
| SIFT + Gabor + GEB | 96.35% | 96.97% |
| SIFT + LBP | 95.70% | 95.20% |
| SIFT + LBP + GEB | 95.62% | 95.24% |
| HOG + LBP | 93.95% | 94.00% |
| HOG + LBP + GEB | 93.88% | 94.10% |
| HOG + Gabor | 96.15% | 95.69% |
| HOG + Gabor + GEB | 96.09% | 95.69% |
| Gabor + LBP | 95.36% | 95.80% |
| Gabor + LBP + GEB | 95.29% | 95.84% |
| SIFT + HOG + LBP | 97.69% | 97.66% |
| SIFT + HOG + LBP + GEB | 97.73% | 97.62% |
| SIFT + Gabor + LBP | 98.50% | 98.05% |
| SIFT + Gabor + LBP + GEB | 98.43% | 97.90% |
| HOG + Gabor + LBP | 97.97% | 97.44% |
| HOG + Gabor + LBP + GEB | 97.91% | 97.39% |
| SIFT + HOG + Gabor | 98.78% | 97.95% |
| SIFT + HOG + Gabor + GEB | 98.78% | 98.01% |
| All four features | **99.48%** | **98.90%** |
| All four features + GEB | **99.44%** | **98.85%** |
| El-Din [24] | 97.11% | - |
| Tapia [106] | 99.13% | 98.01% |
| Alexandre [4] | 99.07% | - |
| Shan [96] | - | 94.81% |

We also conduct similar experiments on the LFW database. Similar to [96], 4,500 males and 2,340 females are chosen. The experimental results are obtained using 5-folds cross-validation. Duplicate images of the same person are placed in the same fold. The experimental results are given in the third column of Table 7.3. It is also observed that using multiple features clearly achieves better accuracy compared to using single features. The negative effect on the recognition accuracy of the GEB could also be neglected. Using all the 4 features, the accuracy is 98.90% without GEB and 98.85% with it. These results are better compared to the methods [96][106] using dense features. In addition, the resulting classifier using all the 4 features with the GEB framework consists of 12 SIFT features, 38 HOG features, 15 Gabor features, and 23 LBP features. The overall feature dimension is around 6,000, which is also much smaller than [96][106]. So using the proposed feature fusion is more discriminative.

Next, we draw the first 8 features selected by the GEB RealAdaBoost algorithm on the LFW database, as shown in Fig. 7.3. There are one SIFT feature, 3 HOG features, 3 Gabor features, and one LBP feature. Only one SIFT feature and LBP feature are selected, which indicates that the GEB selects the features based on both the discriminative ability and the computation cost. In addition, it could be seen that most of the features lay on the upper part of the face. This result is reasonable, because it is much easier to recognize the

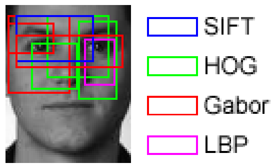


Figure 7.3: The first 8 features selected by the RealAdaBoost with GEB on the LFW database.

Table 7.4: Execution speed of gender classifiers.

| **Approach** | **Time per face(ms)** |
|---|---|
| SIFT | 30.54 |
| HOG | 15.22 |
| Gabor | 20.83 |
| LBP | 12.87 |
| SIFT + HOG | 28.86 |
| SIFT + HOG + GEB | 22.77 |
| SIFT + Gabor | 25.42 |
| SIFT + Gabor + GEB | 18.90 |
| SIFT + LBP | 24.18 |
| SIFT + LBP + GEB | 17.77 |
| HOG + Gabor | 18.93 |
| HOG + Gabor + GEB | 15.49 |
| HOG + LBP | 14.18 |
| HOG + LBP + GEB | 12.43 |
| Gabor + LBP | 17.75 |
| Gabor + LBP + GEB | 14.97 |
| SIFT + HOG + Gabor | 25.62 |
| SIFT + HOG + Gabor + GEB | 18.59 |
| SIFT + Gabor + LBP | 20.11 |
| SIFT + Gabor + LBP + GEB | 14.98 |
| SIFT + HOG + LBP | 19.88 |
| SIFT + HOG + LBP + GEB | 15.64 |
| HOG + Gabor + LBP | 15.10 |
| HOG + Gabor + LBP + GEB | 13.63 |
| All four features | 21.22 |
| All three features + GEB | **12.44** |

gender by eye, eyebrow and nose rather than by mouth, which will be easily influenced by expression variation.

Moreover, we test the resulting gender classifiers on a desktop PC with 3.0 GHZ Intel I7 CPU and 8 GB memory. The execution speed is listed in Table 7.4. We find that SIFT is relatively slow compared to other features. Although LBP is more efficient than others, but the classifier using only LBP requires a large amount of features to achieve a considerable accuracy, so that the overall execution speed of the resulting classifier is not very fast. If we combine the features together and use the GEB framework, the execution time will be reduced, shown as the rows with asterisks. If all four features are used, the speed is significantly improved from 21.22ms per face to 12.44ms per face. So we can get the conclusion that the GEB framework contributes to both the accuracy and the efficiency of the gender recognition.

Table 7.5: Age estimation on PAL and FG-NET database. Units: years old.

| **Approach** | **MAE on PAL** | **MAE on FG-NET** |
|---|---|---|
| SIFT | 5.98 | 5.93 |
| HOG | 6.44 | 6.16 |
| Gabor | 5.88 | 5.68 |
| LBP | 6.98 | 6.54 |
| SIFT + HOG | 5.52 | 5.47 |
| SIFT + HOG + GEB | 5.54 | 5.47 |
| SIFT + Gabor | 5.06 | 5.27 |
| SIFT + Gabor + GEB | 5.05 | 5.24 |
| SIFT + LBP | 5.93 | 5.55 |
| SIFT + LBP + GEB | 5.95 | 5.54 |
| HOG + Gabor | 5.54 | 5.39 |
| HOG + Gabor + GEB | 5.57 | 5.40 |
| HOG + LBP | 5.85 | 5.66 |
| HOG + LBP + GEB | 5.87 | 5.67 |
| Gabor + LBP | 5.82 | 5.24 |
| Gabor + LBP + GEB | 5.82 | 5.27 |
| SIFT + HOG + Gabor | 4.22 | 4.48 |
| SIFT + HOG + Gabor + GEB | 4.24 | 4.49 |
| SIFT + Gabor + LBP | 4.53 | 4.55 |
| SIFT + Gabor + LBP + GEB | 4.55 | 4.58 |
| SIFT + HOG + LBP | 4.85 | 4.70 |
| SIFT + HOG + LBP + GEB | 4.88 | 4.73 |
| HOG + Gabor + LBP | 4.91 | 4.83 |
| HOG + Gabor + LBP + GEB | 4.95 | 4.82 |
| All four features | **4.07** | **4.27** |
| All four features + GEB | **4.09** | **4.28** |
| Choi [16] | 4.33 | 4.66 |
| Kilinc [52] | - | 5.05 |
| Chen [12] | - | 4.67 |

### Age estimation

In this section, we adopt the PAL and FG-NET database to show the effectiveness of the proposed method on age estimation. Similar to gender recognition experiments, the GEB framework consists of the accuracy and efficiency term. For the PAL database, five-folds cross validation is performed, while the age and gender are evenly distributed in each fold. For the FG-NET aging database, Leave-One-Person-Out (LOPO) is utilized because it contains a number of images of the same person. That means, 82-folds are used.

We train 15 age estimators, which includes the classifiers utilizing single feature, and the combination of multiple features. These age estimators consist of two steps, the age group classification and the exact age estimator. The age group classification to classify the ages into 3 groups, child, adult, and seniors; while the exact age estimator is to get the accurate age. The MAE (Mean Absolute Error) of these age estimators on PAL and FG-NET databases are listed in Table 7.5. It can be seen that SIFT and Gabor are more effective compared to HOG and LBP in the age estimation. All the feature combinations including SIFT and Gabor achieve considerable results. This implies that the rotation-invariant gradient and Gabor wavelet are more suitable to describe the age characteristic. Using the feature fusion, the MAE is significantly reduced compared to using single feature. The MAE of the classifiers based on 3 features is similar compared to [16][52][12]. If all 4

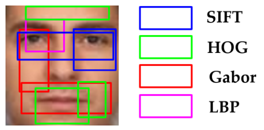


Figure 7.4: The first 8 features selected by the RealAdaBoost with GEB on the FG-NET database.

features are adopted, the MAE is 4.09 on PAL and 4.28 on FERET, which is 0.3 lower than the best result. So it shows the effectiveness of the proposed local feature fusion, compared to [16] based on the global feature, and [52][12] using dense feature vectors. Without the GEB framework, the MAE is only 0.01-0.04 worse. This indicates that the GEB framework will not influence the accuracy much for age estimation either.

The first 8 features selected by the GEB RealAdaBoost for age estimation on the FG-NET database are illustrated in Fig. 7.4, there are 2 SIFT features, 3 HOG features, 2 Gabor features, and one LBP feature. It could be seen that these features lay on the eyes, forehead and mouth region, which is different from the features used for gender recognition in Fig. 7.3. We know that the wrinkles in the forehead or the shape of mouth are more useful in the age estimation rather than in the gender recognition. For example, the wrinkles exist in both male and female's faces, but never in child's faces. So extracting features on these regions is effective for age estimation.

Similarly, we test the resulting age estimators on the same desktop PC. The execution speed is shown in Table 7.6. It could also be seen that the estimators based on SIFT and Gabor are relatively slow compared to HOG or LBP. Using the GEB framework to select features will clearly improve the efficiency. The more features we integrate, due to the fact than using more features will increase the overall discriminative ability so that reduce the feature dimension, the more efficient estimator we will get. If all four features are used, the GEB is able to reduce the execution speed from 23.32ms per face to 13.20ms per face. So the GEB framework is also efficient and effective for age estimation.

## 7.2 Hierarchical weak classifier

We know that a second way to improve the effectiveness in constructing the weak classifier is to integrate multiple feature types. Since different features reflect different characteristic, using more information will lead to better decision. The GEB is able to balance the discrimination ability, generalization power, and computation cost for the same kind of feature. For different kinds of features, GEB only balance the efficiency-accuracy trade-off. In this section, we will introduce our latest work, the hierarchical weak classifier, which solve this trade-off for general feature fusion.

Traditional boosted feature fusion has two major problems. Firstly, the traditional way of feature selection is to evaluate all features before decision. That means, several compu-

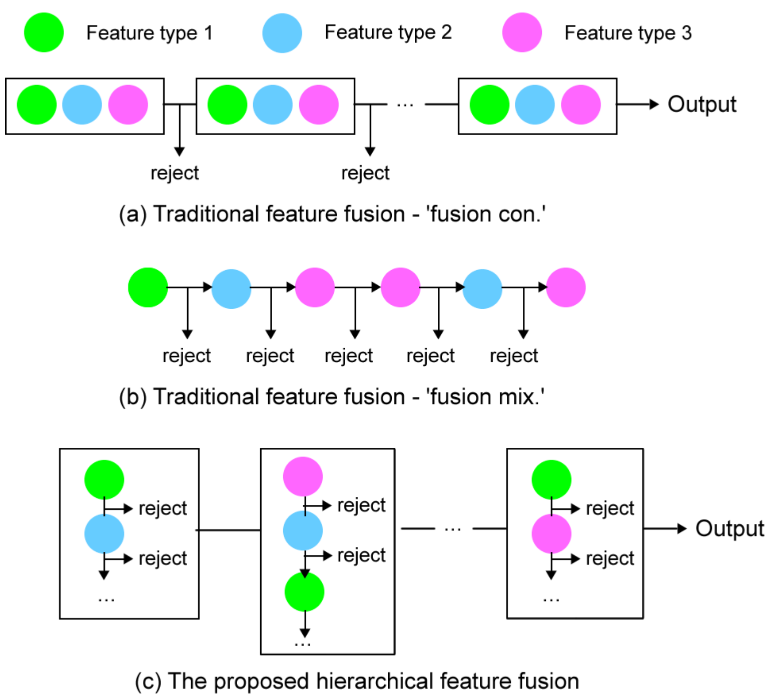
Table 7.6: Execution Speed of age estimators.

| **Approach** | **Time per face(ms)** |
|---|---|
| SIFT | 37.54 |
| HOG | 19.09 |
| Gabor | 26.13 |
| LBP | 16.72 |
| SIFT + HOG | 32.78 |
| SIFT + HOG + GEB | 27.11 |
| SIFT + Gabor | 29.08 |
| SIFT + Gabor + GEB | 24.22 |
| SIFT + LBP | 25.67 |
| SIFT + LBP + GEB | 19.67 |
| HOG + Gabor | 18.09 |
| HOG + Gabor + GEB | 14.94 |
| HOG + LBP | 15.55 |
| HOG + LBP + GEB | 12.24 |
| Gabor + LBP | 20.77 |
| Gabor + LBP + GEB | 16.37 |
| SIFT + HOG + Gabor | 22.08 |
| SIFT + HOG + Gabor + GEB | 17.22 |
| SIFT + Gabor + LBP | 19.65 |
| SIFT + Gabor + LBP + GEB | 15.07 |
| SIFT + HOG + LBP | 18.37 |
| SIFT + HOG + LBP + GEB | 13.67 |
| HOG + Gabor + LBP | 17.00 |
| HOG + Gabor + LBP + GEB | 14.33 |
| All four features | 23.32 |
| All four features + GEB | **13.20** |

tationally expensive feature are always calculated. Secondly, the feature space of different features are also different. For example, the space of the histogram features (e.g. HOG) are different from and the space embedded nonlinear manifold (e.g., covariance matrix). Directly applying the traditional classification techniques based on Euclidean distance will result in the error in other spaces. Therefore, simple concatenation or Cartesian product before the classification is not always effective.

We solve the boosted feature fusion problem by training a detector based on the proposed hierarchical weak classifier. Each weak classifier corresponds to a region of the image and different types of features are extracted from this region. These features are arranged in a hierarchical structure, where each level corresponds to a specific feature type. The classification function for each feature type is learned in its own space. This hierarchical structure weak classifier makes prediction by checking the features level by level. When the confidence of the higher level is not sufficient, the classification will move to the lower level. As illustrated in Fig. 7.5(c), each weak classifier consists of several levels. The decision will start from the first level, and might terminate at any levels. It is different from the tradition ways, such as concatenating all the feature vectors for each weak classifier ("fusion con." in Fig. 7.5(a)) or mixturing different feature types ("fusion mix." in Fig. 7.5(b)). The order in which the features are evaluated is determined based on the GEB measurement in Chapter 7.1.

Compared to traditional feature fusion algorithms, our method has clear advantage. Firstly, we select the hierarchical features considering all the discrimination, the generaliza-



Feature type 1
Feature type 2
Feature type 3
Output
reject
reject
(a) Traditional feature fusion - 'fusion con.'
reject
reject
reject
reject
reject
(b) Traditional feature fusion - 'fusion mix.'
reject
reject
reject
reject
reject
reject
Output
(c) The proposed hierarchical feature fusion

Figure 7.5: Boosted classifier based on hierarchical weak classifier.

tion, and the efficiency. The complicate features are evaluated only when necessary. The resulting boosted classifier performs well on both the accuracy and the efficiency in real object detection task. Secondly, the classification functions of different types of features are learned in their own spaces. The features are organized in a hierarchical way, so that we do not need to consider the concatenation or normalization of these feature vectors.

## 7.2.1 Weak Classifier Design

### Overview

Boosting selects a series of weak classifiers to construct the strong classifier. Each weak classifier corresponds to a specific image region $R$. Suppose $k$ different types of local features $\{f_1, \ldots, f_k\}$ are extracted from this region, denote the weak classifier by $h$, which could be considered as a mapping from the input space $X$ to a confidence space. The output of $h$ indicates the predicted class, positive for object and negative for non-object. The absolute value represents the classification confidence. Denote the feature space by $F$, given an input sample $\mathbf{x}$, the weak classifier based on the subset of features $\{f_1, \ldots, f_k\}$ could be formulated as

$$h(\mathbf{x}) = h(f_1, \ldots, f_k, \mathbf{x}) = h(S(\mathbf{x}), \mathbf{x}) \tag{7.7}$$

where $S : X \to F$ is a feature type selector.

In traditional boosting methods, the only objective is to maximize the accuracy. In our case, we also consider the generalization power and computation cost in the feature selection, which are important factors in real object detection systems. We directly use our previous GEB framework as the feature evaluation score.

Our weak classifier is approximated by a combination of several single-feature classifiers, each of which is based on one feature type. The computations of different features are independent. We want to use the subset of features with the highest GEB score

$$S^*(\mathbf{x}) = \underset{\{f_1,\dots,f_k\}\in F}{argmax} \{GEB(h(f_1,\dots,f_k,\mathbf{x}))\}. \tag{7.8}$$

Then the expected GEB score of $h(S^*(\mathbf{x}),\mathbf{x})$ on $X$ is

$$E = \sum_{i=1}^{k} P(S^* = F_i) E(GEB(h(F_i,\mathbf{x}))|S^* = F_i), \tag{7.9}$$

where $F_i = \{f_1,\dots,f_i\}$.

## Construction of Hierarchical Weak Classifier

In the implementation, it is impossible to compute $S^*$, since it needs to know the best features before evaluating the images. In our work, we propose a hierarchical weak classifier to approximate $h(S^*(\mathbf{x}),\mathbf{x})$. This weak classifier $h(f_1,\dots,f_k,\mathbf{x}) = h(F_k,\mathbf{x})$ consists of $k$ levels, where each level $h_i, i = 1,\dots,k$ corresponds to one feature type. It could be defined recursively

$$h(F_i,\mathbf{x}) = \begin{cases} h(F_{i-1},\mathbf{x}), & if \mathbf{x} \in C(i-1,X) \\ h(f_i,\mathbf{x}), & otherwise \end{cases}, \tag{7.10}$$

where $h(f_i,\mathbf{x})$ is the weak classifier based on the $i$th feature, $C(i-1,X)$ is the set of samples where the prediction of the previous levels $h(F_{i-1},\mathbf{x})$ is confident, defined as

$$C(i,X) = C(i-1,X) \bigcup \{\mathbf{x}|\beta_i < |h(f_i,\mathbf{x})|\}. \tag{7.11}$$

$\beta_i$ is a confidence threshold for feature $f_i$

$$\beta_i = \underset{\beta}{argmin}\, P(\beta < |h(f_i,\mathbf{x})|) \leq \frac{P(S^* = F_i)}{1 - \sum_{j=1}^{i-1} P(S^* = F_j)}. \tag{7.12}$$

If the feature types used are fully independent, $P(S^* = F_i) = P(\mathbf{x} \in \{C(i,X) - C(i-1,X)\})$. Denote $C(i) = C(i,X) - C(i-1,X)$, if $\mathbf{x} \in C(i)$, then $S = F_i = \{f_1,\dots,f_i\}$, and $h(S,\mathbf{x}) = h(F_i,\mathbf{x}) = h(f_1,\dots,f_i,\mathbf{x})$. The expected GEB of the hierarchical weak classifier in Eq. 7.9 will be equalized to

$$E = \sum_{i=1}^{k} P(\mathbf{x} \in C(i)) E(GEB(h(F_i, \mathbf{x}) | \mathbf{x} \in C(i))). \tag{7.13}$$

Using Eq. 7.13, the classification power of the hierarchical weak classifier with multiple features is measured by its expected GEB. At each boosting round, we evaluate several regions. For each of them, we find the best feature of each feature type and combine them to form $h(F_k, \mathbf{x})$. The one with the largest expected GEB is added to the current cascade classifier.

### 7.2.2 Implementation Details

#### Features

We use four features to construct the hierarchical weak classifier, including LBP, HOG, COV, and CoHOG.

The HOG, LBP, CoHOG are the same as mentioned in previous chapters. The COV is extracted from a 7-D raw feature vector: $\{x, y, I, |d_x|, |d_y|, |d_{xx}|, |d_{yy}|\}$, where x and y are the pixel location, $I$ is the intensity, and $|d_x|, |d_y|, |d_{xx}|, |d_{yy}|$ are the first/second order intensity derivatives. The covariance matrices lie in a connected Riemannian manifold [109]. Using similar method as [109], we first map the covariance matrices to a linear tangent space of the manifold, and then perform regularized least square in the tangent space to map the feature vector to single dimension. The dimension of the tangent vector is 28.

#### Boosting learning

The details of the boosted learning based on hierarchical weak classifiers are illustrated in Fig. 7.6. In each iteration, we randomly sample $M = 50$ regions and search for the locally best LBP, HOG, COV, and CoHOG features. For each region $R$, the local search is done by randomly evaluating $O = 20$ features for each feature type. These features should cover at least 50% of $R$. The feature vectors are calculated and mapped to one dimension. Then the $W_+$ and $W_-$ are calculated to build the single weak classifier for each candidate feature. After getting the best feature $f_1, \ldots, f_k$ for each feature type, the hierarchical weak classifier $h(f_1, \ldots, f_k, \mathbf{x})$ is constructed along with estimating $\beta_k$ and $C(k)$ level by level. The one with the largest GEB score will be selected to construct the strong classifier.

The computational costs of the four feature types are different. Computing a LBP is based on binary matching, while the HOG extraction requires caculating the gradient magnitude and orientation to generate the integral image. Computing covariance matrix requires singular value decomposition. After applying experiments on several computers, we set the computational cost $t$ of these four features to 1, 2, 10, 5 respectively.

The parameters $\alpha, \beta$ in the GEB score are determined by several experiments on PASCAL VOC 2007 and 2010 datasets. In our final implementation, we set $\alpha = 0.15, \beta = 0.2$.

```
Parameters
  N    number of training samples
  M    number of evaluated regions each iteration
  K    number of feature types
  O    number of random evaluated features for each feature type
  T    maximum number of weak classifiers
Input: Training set $\{(\mathbf{x}_i, y_i)\}, y_i \in \{-1, 1\}$
1. Initialize sample weight and classifier output
  $w_i = 1/N, H(\mathbf{x}_i) = 0$
2. Repeat for $t = 1, 2, \ldots, T$
  2.1 Update the sample weight $w_i$ using the $t^{th}$ weak classifier output
    $w_i = w_i e^{-y_i h_t(f_{t,1}, \ldots, f_{t,k}, \mathbf{x}_i)}$
  2.2 For $m = 1$ to M
      2.2.1 Generate a random region $R$
      2.2.2 For $k = 1$ to K
              For $o = 1$ to O
        2.2.2.1 Generate a random LBP, HOG, COV, and CoHOG
        2.2.2.2 Calculate the feature vectors and map to single dimension
        2.2.2.3 Select the one with best GEB score to construct single
                weak classifier $h_t(f_k, \mathbf{x})$
      2.2.3 For $k = 1$ to K
        2.2.3.1 Estimate $\beta_k$ and $C(k)$
        2.2.3.2 Add $h_t(f_k, \mathbf{x})$ to the hierarchical weak classifier as $h_t(f_1, \ldots, f_k, \mathbf{x})$
        2.2.3.3 Calculate expected GEB score of the hierarchical weak classifier
  2.3 Choose the hierarchical weak classifier with the maximum expected GEB score
  2.4 Update strong classifier $H_{t+1}(\mathbf{x})$
  2.5 Update fp. If the fp is lower than $10^{-6}$, break
3. Output classifier $H(\mathbf{x}) = sign[\sum_{j=1}^{t} h_j(\mathbf{x})]$
```

Figure 7.6: Boosting training with hierarchical weak classifiers.

The first bootstrap will be called when the current false positive rate is lower than 0.5. Then all the filtered negative samples will be removed, and new negative samples will be used to fill this gap. Every time when the false positive rate achieves 0.5 or lower, the bootstrap will be applied. The whole training will terminate when the overall false positive rate is lower than $10^{-6}$ or the weak classifier number exceeds the maximum $T$.

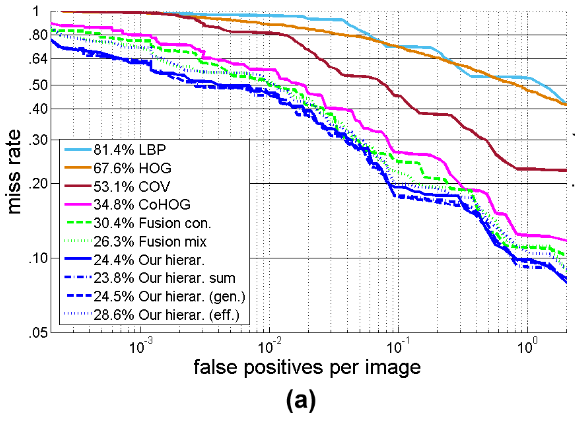
### 7.2.3 Experimental Results

#### Experiments on Caltech dataset

Firstly, we evaluate our method using the Caltech pedestrian dataset. We follow the training and evaluation protocol proposed by Dollar et al. [23]. The pedestrians at least 50 pixels tall under no or partial occlusion are evaluated. Fig. 7.7(a) illustrates the experimental results of our methods compare to using single feature and traditional feature fusion methods. It could be seen that compared to using single LBP, HOG, COV, and CoHOG, our feature fusion achieves significant better performance. We also compare our method with other two traditional feature fusion strategies, labeled as "fusion concatenation (fusion con.)" and "fusion mix". The "fusion concatenation" is to directly concatenating all feature vectors and then project them to single dimension. The "fusion mix" is to mix all the feature pools for LBP, HOG, COV, and CoHOG, and select only one of them for each weak classifier. The results show that our hierarchical classifier clearly outperforms these two fusion strategies. Since different features lay on different distance space, simply concatenating them is not a good idea. The "fusion mix" achieves 2% lower accuracy compared to our hierarchical solution, and the efficiency is also 6 times lower. This result is reasonable, because the "fusion mix" fails to consider both the generalization and computation cost in the training procedure.

In addition, instead of using our level-by-level evaluation strategy, we use all features in the hierarchical weak classifier for decision (labelled as "our hierar. sum" in the figures). This is achieved by summing all the output of the features in our hierarchical classifier. It can be considered as the accuracy upper bound of our hierarchical strategy. From Fig. 7.7(a) we can find that it is slightly better than our hierarchical decision. This indicates that the proposed method based on GEB feature selection and hierarchical decision is a good approximation. The efficiency of summing all of them will be 8 times lower compared to our solution.

In Fig. 7.7(a), the last two curves are utilized to show the effectiveness of GEB. Using the same feature pool, we train two detectors while keeping the generalization term but removing the efficiency penalty term (Our hierar. gen.), and keeping the efficiency penalty term but removing the generalization term (Our hierar. eff.). It could be seen that without the generalization term, the accuracy is reduced 4.1%, which indicates that considering the generalization in the training process will contribute to the accuracy. In addition, adding



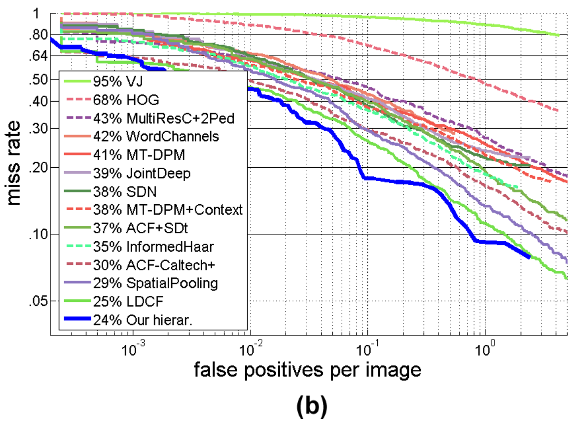


Figure 7.7: FPPI evaluation on Caltech pedestrian dataset. (a) Comparison of the boosted classifiers based on different features and feature fusions. (b) Comparison to other methods.

Table 7.7: Frequencies of different feature types as the first, second, third, and fourth level in the hierarchical weak classifiers.

| **Feature** | LBP | HOG | COV | CoHOG |
|---|---|---|---|---|
| First level | 34% | 47% | 2% | 17% |
| Second level | 14% | 27% | 14% | 45% |
| Third level | 9% | 30% | 22% | 39% |
| Fourth level | 5% | 27% | 38% | 30% |

the computation penalty will not influence the accuracy much. So the GEB evaluation is effective.

In Fig. 7.7(b), we compare our results with the commonly-used algorithms on pedestrian detection. We notice that using the hierarchical boosted classifier, the accuracy is much better than other boosting family methods (Multiftr, FPDW, pAUCBoost, FisherBoost, Crosstalk, SketchTokens, Spatialpooling, LDCF, ACF-Caltech+). So the proposed hierarchical feature fusion framework achieves high accuracy.

### 7.2.4 Feature analysis

Here we analyze the trained detectors on Caltech database. Firstly, we count the frequencies of different types of features selected as the first/second/third/fourth level in the weak classifiers, shown in Table 7.7. It can be seen that though COV and CoHOG are stronger than HOG and LBP for classification, they are much more computationally expensive so that they are mostly used as the lower level in the hierarchical weak classifier.

Next, we count the evaluation frequencies for the four levels in the hierarchical weak classifiers. In the detection procedure, each $64 \times 128$ window will evaluate 22.76 first level features, 4.67 second level features, 1.19 third level features, and 0.44 fourth level features. Since the first features are mostly LBP and HOG, the evaluation process will be relatively efficient.

Table 7.8: Testing speed of the boosted classifiers with different feature and feature fusion methods.

| **Approach** | **Patches/sec** |
|---|---|
| HOG | 247k |
| LBP | 377k |
| COV | 40k |
| CoHOG | 77k |
| fusion con. | 24k |
| fusion mix | 38k |
| Ours hierar. | 224k |
| Ours hierar. sum | 27k |

Table 7.9: Experimental results of the hierarchical weak classifier on PASCAL VOC 2007 dataset.

| **VOC2007 test** | aero | bike | bird | boat | bot. | bus | car | cat | cha. | cow | tab. | dog | hor. | mot. | per. | pla. | she. | sofa | tra. | tv | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HOG | 37.8 | 52.3 | 19.9 | 17.2 | 18.3 | 31.0 | 47.9 | 23.5 | 18.3 | 23.2 | 22.1 | 19.2 | 39.0 | 46.0 | 30.1 | 17.2 | 23.4 | 24.4 | 42.1 | 38.3 | 29.6 |
| LBP | 26.8 | 41.8 | 18.4 | 16.3 | 17.2 | 29.5 | 47.4 | 22.5 | 17.1 | 22.4 | 20.1 | 19.2 | 36.0 | 35.0 | 31.1 | 16.1 | 22.5 | 18.0 | 41.5 | 37.3 | 26.8 |
| COV | 30.0 | 47.0 | 23.1 | 21.5 | 22.0 | 35.4 | 52.2 | 28.8 | 24.2 | 25.5 | 25.1 | 23.9 | 42.7 | 39.8 | 37.1 | 22.3 | 26.1 | 23.4 | 49.1 | 41.3 | 32.0 |
| CoHOG | 32.7 | 47.4 | 23.9 | 23.3 | 24.2 | 38.2 | 55.2 | 30.8 | 23.4 | 27.2 | 27.2 | 26.7 | 42.5 | 40.6 | 42.9 | 22.3 | 29.1 | 25.9 | 50.0 | 44.3 | 33.9 |
| fusion con. | 45.5 | 53.9 | 25.2 | 23.1 | 25.7 | 37.2 | 57.1 | 32.0 | 25.9 | 28.2 | 32.1 | 27.6 | 44.4 | 43.3 | 42.1 | 25.3 | 29.8 | 30.5 | 51.0 | 45.8 | 36.1 |
| fusion mix | 53.8 | 59.5 | 30.3 | 29.3 | 25.9 | 42.8 | 57.9 | 30.5 | 25.3 | 30.5 | 36.1 | 31.2 | 49.3 | 56.0 | 46.1 | 29.2 | 32.5 | 34.4 | 51.2 | 45.3 | 39.9 |
| Our hierar. | 61.3 | 58.0 | 31.3 | 28.9 | 30.5 | 51.1 | 63.1 | 45.7 | 26.0 | 34.4 | 46.5 | 36.4 | 57.4 | 60.1 | 47.3 | 26.9 | 36.4 | 40.6 | 50.3 | 52.6 | **44.2** |
| Our hierar. sum | 61.9 | 58.4 | 31.1 | 29.4 | 29.3 | 50.3 | 63.1 | 45.2 | 26.6 | 34.4 | 46.3 | 35.5 | 57.2 | 62.4 | 51.3 | 27.2 | 36.2 | 40.6 | 52.4 | 53.3 | **44.6** |
| Chen [11] | 41.0 | 64.3 | 15.1 | 19.5 | 33.0 | 57.9 | 63.2 | 27.8 | 23.2 | 28.2 | 29.1 | 16.9 | 63.7 | 53.8 | 47.1 | 18.3 | 28.1 | 42.2 | 53.1 | 49.3 | 38.7 |
| Cinbis [17] | 56.1 | 56.4 | 21.8 | 26.8 | 19.9 | 49.5 | 57.9 | 46.2 | 16.4 | 41.4 | 47.1 | 29.2 | 51.3 | 53.6 | 28.6 | 20.3 | 40.5 | 39.6 | 53.5 | 54.3 | 40.5 |
| Wang [118] | 54.2 | 52.0 | 20.3 | 24.0 | 20.1 | 55.5 | 68.7 | 42.6 | 19.2 | 44.2 | 49.1 | 26.6 | 57.0 | 54.5 | 43.4 | 16.4 | 36.6 | 37.7 | 59.4 | 52.3 | 41.7 |
| Felz. [28] | 36.6 | 62.2 | 12.1 | 17.6 | 28.7 | 54.6 | 60.4 | 25.5 | 21.1 | 25.6 | 26.6 | 14.6 | 60.9 | 50.7 | 44.7 | 14.3 | 21.5 | 38.2 | 49.3 | 43.6 | 35.4 |

Moreover, we test the detection speed of the above detectors on an Intel I7 dual core PC with 8GB memory. The results are listed in Table 7.8. It could be seen that the testing speed of the hierarchical boosted classifier is far better compared to the traditional feature fusion algorithms (fusion con., fusion mix). Compared to using single COV or CoHOG, the efficiency is also better. Although the sum of all hierarchical features is able to achieve slightly better accuracy, the speed is 8 times slower. So we can get the conclusion that the propose method also achieves considerable efficiency.

### Experiment on PASCAL dataset

Next, we utilize the PASCAL VOC 2007 and 2010 dataset to test our method on general object detection. We follow the requirement of the training set and testing set of the PASCAL

Table 7.10: Experimental results of the hierarchical weak classifier on PASCAL VOC 2010 dataset.

| **VOC2010 test** | aero | bike | bird | boat | bot. | bus | car | cat | cha. | cow | tab. | dog | hor. | mot. | per. | pla. | she. | sofa | tra. | tv | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HOG | 40.6 | 30.9 | 12.1 | 11.7 | 15.7 | 32.3 | 44.6 | 23.1 | 10.1 | 13.1 | 11.7 | 16.5 | 29.3 | 38.3 | 29.4 | 8.3 | 22.9 | 18.1 | 33.6 | 26.2 | 24.9 |
| LBP | 24.2 | 22.8 | 18.4 | 7.0 | 11.9 | 28.4 | 43.1 | 20.7 | 5.8 | 12.8 | 7.3 | 11.9 | 24.9 | 27.4 | 22.3 | 6.7 | 14.5 | 12.2 | 29.4 | 22.3 | 20.1 |
| COV | 28.5 | 27.5 | 13.7 | 11.5 | 12.9 | 34.1 | 49.4 | 27.5 | 11.2 | 15.8 | 12.6 | 16.9 | 31.4 | 30.3 | 26.5 | 9.3 | 23.8 | 20.0 | 34.1 | 26.3 | 24.8 |
| CoHOG | 34.3 | 37.7 | 13.4 | 16.3 | 18.4 | 38.5 | 53.7 | 31.8 | 10.4 | 20.7 | 17.2 | 19.6 | 35.6 | 34.1 | 35.1 | 9.3 | 29.0 | 24.2 | 38.3 | 32.3 | 29.3 |
| fusion con. | 47.7 | 40.7 | 16.4 | 19.2 | 21.4 | 41.5 | 47.0 | 35.5 | 14.3 | 23.1 | 20.6 | 24.3 | 39.1 | 48.6 | 37.0 | 13.1 | 31.7 | 24.3 | 37.0 | 30.5 | 32.5 |
| fusion mix | 55.4 | 47.2 | 23.5 | 26.0 | 27.6 | 45.9 | 54.5 | 41.1 | 22.0 | 25.7 | 25.4 | 32.5 | 44.8 | 53.1 | 40.3 | 17.9 | 36.4 | 30.4 | 43.8 | 35.9 | 38.2 |
| Our hierar. | 62.7 | 52.2 | 24.3 | 27.0 | 26.0 | 55.3 | 56.2 | 48.5 | 20.7 | 30.7 | 33.7 | 37.5 | 50.2 | 57.3 | 45.9 | 14.0 | 39.9 | 33.3 | 51.3 | 41.9 | **42.9** |
| Our hierar. sum | 63.3 | 53.9 | 26.1 | 27.4 | 26.8 | 57.2 | 56.2 | 49.2 | 20.4 | 30.4 | 34.2 | 38.3 | 49.9 | 58.5 | 47.3 | 14.2 | 41.0 | 34.7 | 51.0 | 45.3 | **43.7** |
| Chen [13] | 54.6 | 53.7 | 16.2 | 12.5 | 31.2 | 54.0 | 44.2 | 40.0 | 16.7 | 32.2 | 29.1 | 30.1 | 54.3 | 57.2 | 43.9 | 12.5 | 35.4 | 28.8 | 51.1 | 40.7 | 36.9 |
| Everingham [27] | 56.7 | 39.8 | 16.8 | 12.2 | 13.8 | 44.9 | 36.9 | 47.7 | 12.1 | 26.9 | 26.5 | 37.2 | 42.1 | 51.9 | 25.7 | 12.1 | 37.8 | 33.0 | 41.5 | 41.7 | 32.9 |
| Wang [118] | 65.0 | 48.0 | 25.9 | 24.6 | 24.5 | 56.1 | 54.5 | 51.2 | 17.0 | 28.9 | 30.2 | 35.8 | 40.2 | 55.7 | 43.5 | 14.3 | 43.9 | 32.6 | 54.0 | 45.9 | 39.7 |
| Felz. [28] | 49.2 | 53.8 | 13.1 | 15.3 | 35.5 | 53.4 | 49.7 | 27.0 | 17.2 | 28.8 | 14.7 | 17.8 | 46.4 | 51.2 | 47.7 | 10.8 | 34.2 | 20.7 | 43.8 | 38.3 | 33.4 |

challenge [27]. These size are utilized in both the VOC 2007 and VOC 2010 experiments. In Table 7.9 and 7.10, we compare our method with the top methods using conventional machine learning methods on PASCAL VOC 2007 and 2010, in terms of detection AP on the test set. Firstly, it could be seen that the mAP of our feature fusion is 0.442 on VOC 2007 and 0.429 on VOC 2010, which is better compared to using single feature. So combining boosted features is also effective for general object detection. In addition, we notice that the our hierarchical feature fusion still works better compared to concatenation (fusion con.) and mixing (fusion mix.). It is also a good approximation compared to the optimal case (our hierar. sum). Compared to other methods with conventional learning algorithms, our hierarchical boosted classifier is more effective than pyramid HOG [28], HOG+LBP+Context [13], MOCO [11], and SIFT fisher vectors [17]. Our accuracy is also better than heterogeneous features [118], which is a published work on using multi-scale HOG and COV fusion.

## 7.3 Conclusion

In this chapter, two methods dealing with the efficiency-accuracy trade-off problem in boosting training are introduced. The Generalization and Efficiency Balanced (GEB) framework considers all the discriminative ability, generalization power, and computation cost in the feature selection procedure. The boosting algorithm will emphasize the efficient and robust features in the beginning stages. As a result, we may get the boosted classifier both efficient to compute and robust to noise. The hierarchical weak classifier organizes different types of features in a hierarchical way. The features are ranked according to the GEB protocol, which ensures the robustness and efficiency. Computationally expensive features are only used if the confidence of previous features are not sufficient enough. The resulting boosted classifiers show good performance on the general object detection task.

# Chapter 8

# Conclusion and future work

## 8.1 Conclusion

Object detection is an indispensable technology in many applications such as artificial intelligence systems, mobile devices, and video surveillance. This thesis summarizes the difficulties in object detection into two aspects, the large intra-class variations and low SNR. Large intra-class variation means that the object shape and appearance in the same category are very different. Low SNR means that the useful object information is very limited compared to the noise. To overcome these two difficulties, we work on the local features and boosting methods respectively.

To solve the large intra-class variations problem, we focus on designing more discriminative and robust local features. We organize the local features into a "feature hierarchy". Different strategies are applied for the features in different levels. For the low-level features, we focus on how to extract the binary information. For mid-level features, we work on improving the discriminative ability of the gradient features. The local structural information will be integrated in addition to the gradient information. For high-level features, we generalize a kind of co-occurrence patterns, which show promising accuracy in general object detection.

To solve the low SNR problem, we work on the boosted classifiers. Firstly, we try to enhance the weak classifier learning procedure. Since the discriminative ability of a boosted classifier is decided by the weak classifier, strengthening the weak classifier learning will contribute to the overall accuracy. In addition, we design two methods to balance the accuracy and efficiency in boosting training. As a result, we may get the boosted classifiers that not only have high accuracy, but also achieve good efficiency.

More specifically, the contribution of this thesis could be summarized as follows:

1. For low-order features, we propose the Boosted Local Binaries (BLB) [81] and Boosted Binary Patterns (BBP) [83]. These two features use a series of binary patterns with variable

size and location to represent the target object. Experimental results show that these two features show good performance for object detection and tracking.

2. For mid-order features, we first introduce Edged Histogram of Oriented Gradient feature (Edge-HOG) [82]. Edge-HOG extracts the gradient vector from a series of regions. These regions are arranged along an edge to represent the local structural information. The experimental results show that the Edge-HOG achieves better accuracy compared to traditional gradient features. In addition, we design the Deformable Edge Set (DES) [84], which is a combination of several short contour segments. Each segment is deformed from the edge template to the actual object contour according to the distribution model of pixel gradients. Experimental results show that the proposed DES is good at detecting the objects with clear contours.

3. For high-order features, we design a kind of co-occurrence features [85]. The co-occurrence features are defined as the distribution of co-occurring variables at a given offset. These variables are constructed by low-order information such as intensity or gradient. Using different low-order information, the co-occurrence features are able to capture the key characteristics of different object categories. Experimental results show that the co-occurrence features outperform other traditional features in pedestrian detection and general object detection.

4. In order to enhance the weak classifier learning, we propose the basis mapping approach [80]. Basis mapping is a non-linear mapping on original samples by referring to the basis samples before learning the weak classifiers, where the basis samples correspond to the hard samples in the current training stage. We show that the basis mapping based weak classifier is an approximation of kernel weak classifier while keeping the same computation cost as linear weak classifier. Experimental results show that the basis mapping consistently improves the detection accuracy and training efficiency of the boosted classifier.

5. In order to deal with the efficiency-accuracy trade-off, we first design the the Generalization and Efficiency Balanced (GEB) framework [85]. In the feature selection procedure, the discriminative ability, the generalization power, and the computation cost of the candidate features are all evaluated for decision. In addition, we propose the hierarchical weak classifier for feature fusion. The weak classifier makes predictions by examining the features level by level. Highly computational features are used if the efficient features could not make a confident decision. As a result, we could get the boosted detectors with both high accuracy and good efficiency.

## 8.2 Discussion

As mentioned in Chapter 3, deep network is one of the hottest topics in current object detection. Deep learning algorithms are able to achieve better accuracy compared to traditional learning algorithms. However, the research on traditional features and machine learning algorithms are still important.

Firstly, as mentioned in the LeCun's talk in CVPR 2015 [56], deep learning has a long way to go in the theoritical part. For example, although deep networks achieve high accuracy, it is still difficult to explained why it is better compared to SVM or AdaBoost. It is also not clear if a deepnet has converged or not [112]. In contrast, we have a good understanding of how to model multiple modalities with traditional tools. That is currently an active field of research in the deep learning land.

Secondly, there are many parameters for the neural network, and they are difficult to interpret. Although this case occurs in some traditional learning algorithms, the difficulty is far smaller compared to the one in the deep learning. As a result, it is very difficult to debug the deep networks. For example, in most of the time, it is not clear about how many layers and free parameters we really need. The only way is to try different parameters based on some empirical heuristics. In addition, The scale of a deepnet's weights (and of the weight updates) is also very important for performance. When the features are of the same type (pixels, word counts, etc.), this is not a problem. When the features are heterogeneous (e.g., text+image), we need to standardize the inputs in some way. This also motivates the research on traditional features.

Thirdly, deep networks do not show promising performance when the size of the dataset is small, e.g., less than 10k. Larger dataset means long training time and long period before seeing the result. So it might not satisfy the requirement of some applications, especially for some industrial projects with short periods. In addition, many problems do not need the fire power of a deep neural network. For example, the usage of CCA for OCR has already been commercialized for 20+ languages. Using binary features with AdaBoost could achieve 99% detection rate for the faces less than 30 in-plan rotation degrees. In these cases, the traditional features and learning methods will be more useful.

We also consider the relationship of our research and deep learning. On one hand, deep networks try to derive the high-level information based on the traditional features. The resulting deep features could be considered as the refined version of a super feature pool. Some current deep networks use raw images as input. It might be better to use both the raw image and some feature maps. Utilizing more discriminative traditional features has the potential to further improve the accuracy. As mentioned in [85], the proposed features explore the co-occurrence information explicitly, while the convolution networks use such information implicitly. On the other hand, after we get the deep features, traditional machine learning algorithms such as SVM or Boosting are applied to generate the final classification result. Our research on Boosting further improves the performance of these algorithms.

## 8.3 Future work

While we have provided multiple contributions on local features, much more can be done to further improve the accuracy and robustness. Firstly, the proposed co-occurrence features are designed as the co-occurrence matrix for the information at the same level, e.g., gradient versus gradient, intensity versus intensity. We have not discussed how to extract the co-occurrence information between different levels, e.g., gradient versus intensity. Our latest experimental results show that such co-occurrence information is a good complementary to the one based on the same level. Further studying this topic seems to be a good idea. In addition, we have not investigated how to fuse the local features from different levels in the feature hierarchy, e.g., how to combine the binary features with gradient features. Some prior work only use the feature fusion as prior knowledge (e.g., HOG+LBP, HOG+COV), without further explanation. It might be better to analyze why fusing these features is good. Our hierarchical weak classifier fuse these features from the view of balancing the accuracy-efficiency trade-off. It will be better if we could find the reason from the context or semantic level.

Our research on machine learning is designed for boosting framework, we also consider to apply similar ideas to other classifiers. For example, the idea of the proposed basis mapping is due to the fact that boosting emphasizes the difficult samples. There are other classifiers doing similar things, such as in the support vector machine, where the support vectors are the samples closet to the classification plane. These samples are also considered as the difficult samples. In addition, the efficiency-accuracy trade-off is a classic problem in any machine learning tasks. Extending them to other classifiers such as deep networks is a valuable topic.

Moreover, our work is applicable not only for object detection, but also for other applications. Since most of the tasks in the computer vision share the same feature, the proposed local features could also be applied to other applications. For example, the BBP feature is a super set of BLB feature, so it is effective for both object detection and object tracking. The co-occurrence feature is also proved to work well in face recognition and food recognition. Regarding the boosting, most of our research does not change the infrastructure. It will not influence the error function and weight update, which is the essential of boosting convergence. That means, our research is able to be applied for other boosting algorithms. In this thesis, we have already shown that how to use RealAdaBoost and LogitBoost with our work. It could be generalized to other boostings such as GentleBoost. In addition, our work might be suitable for other applications using boosting algorithm. We have already proven the effectiveness of our work on object detection, object tracking, gender recognition, and age estimation. It is possible to investigate more potential applications.

# Bibliography

[1] Pulkit Agrawal, Ross Girshick, and Jitendra Malik. Analyzing the performance of multilayer neural networks for object recognition. In *European Conference on Computer Vision*, pp. 329-344, 2014.

[2] Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 26, pp. 1475–1490, 2004.

[3] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. In *IEEE Transaction on Pattern Analysis of Machine Intelligence*, Vol. 28, pp. 2037–2041, 2006.

[4] Luís Alexandre. Gender recognition: A multiscale decision fusion approach. In *Pattern Recognition Letters*, Vol. 31, pp. 1422–1427, 2010.

[5] Pablo Arbeláez, Jordi Pont-Tuset, Jonathan Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 329-335, 2014.

[6] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Robust object tracking with online multiple instance learning. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, pp. 1619–1632, 2011.

[7] Aharon Bar-Hillel, Dan Levi, Eyal Krupka, and Chen Goldberg. Part-based feature synthesis for human detection. In *European Conference on Computer Vision*, pp. 127-142, 2010.

[8] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision*, pp. 404-417, 2006.

[9] Rodrigo Benenson, Markus Mathias, Radu Timofte, and Luc Van Gool. Pedestrian detection at 100 frames per second. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2903-2910, 2012.

[10] Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: A review and new perspectives. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 35, pp. 1798–1828, 2013.

[11] Guang Chen, Yuanyuan Ding, Jing Xiao, and Tony Han. Detection evolution with multi-order contextual co-occurrence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1798-1805, 2013.

[12] Ke Chen, Shaogang Gong, Tao Xiang, and Chen Loy. Cumulative attribute space for age and crowd density estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2467–2474, 2013.

[13] Qiang Chen, Zheng Song, Jian Dong, Zhongyang Huang, Yang Hua, and Shuicheng Yan. Contextualizing object detection and classification. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, pp. 13–27, 2015.

[14] Ming Cheng, Niloy Mitra, Xumin Huang, Philip Torr, and Song Hu. Global contrast based salient region detection. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 37, pp: 569–582, 2015.

[15] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3286-3293, 2014.

[16] Sung-Eun Choi, Youn-Joo Lee, Sung-Joo Lee, Kang-Ryoung Park, and Jaihie Kim. Age estimation using a hierarchical classifier based on global and local facial features. In *Pattern Recognition*, Vol. 44, pp. 1262–1281, 2011.

[17] Ramazan Cinbis, Jakob Verbeek, and Cordelia Schmid. Segmentation driven object detection with fisher vectors. In *IEEE International Conference on Computer Vision*, pp. 2968-2975, 2013.

[18] Arthur Costea and Sergiu Nedevschi. Word channel based multiscale pedestrian detection without image resizing and using only one classifier. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2393-2400, 2014.

[19] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 886-893, 2005.

[20] Piotr Dollár, Ron Appel, Serge Belongie, and Pietro Perona. Fast feature pyramids for object detection. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 36, pp. 1532–1545, 2014.

[21] Piotr Dollár, Ron Appel, and Wolf Kienzle. Crosstalk cascades for frame-rate pedestrian detection. In *European Conference on Computer Vision*, pp. 645-659, 2012.

[22] Piotr Dollár, Serge Belongie, and Pietro Perona. The fastest pedestrian detector in the west. In *British Machine Vision Conference*, pp. 1-7, 2010.

[23] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. In *IEEE Transaction on Pattern Analysis of Machine Intelligence*, Vol. 34, pp. 743–761, 2012.

[24] Yomna El-Din, Mohamed Moustafa, and Hani Mahdi. Landmarks-SIFT Face Representation for Gender Classification. In *International Conference on Image Analysis and Processing*, pp. 329–338, 2013.

[25] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2155-2162, 2014.

[26] Ali Eslami, Nicolas Heess, Christopher Williams, and John Winn. The shape boltzmann machine: a strong model of object shape. In *International Journal on Computer Vision*, Vol. 107, pp, 155–176: 2014.

[27] Mark Everingham, Luc Van Gool, Christopher Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88:303–338, 2010.

[28] Pedro Felzenszwalb, Ross Girshick, David McAllester, and Dev Ramanan. Object detection with discriminatively trained part-based models. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 32, PP: 1627–1645, 2010.

[29] Vittorio Ferrari, Loic Fevrier, Cordelia Schmid, and Frédéric Jurie. Groups of adjacent contour segments for object detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, pp. 36–51, 2008.

[30] Vittorio Ferrari, Frederic Jurie, and Cordelia Schmid. From images to shape models for object detection. In *International journal of computer vision*, Vol. 87, pp. 284–303, 2010.

[31] FGNET face database. http://www.fgnet.rsunit.com.

[32] Yoav Freund and Robert Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pp. 148-156, 1996.

[33] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors. In *Annals of Statistics*, Vol. 28, pp. 337-407, 2000.

[34] Ross Girshick. Fast R-CNN. In *arXiv:1504.08083v1*, 2015.

[35] Georgia Gkioxari, Bharath Hariharan, Ross Girshick, and Jitendra Malik. R-CNNs for pose estimation and action detection. In *arXiv:1406.5212*, 2014.

[36] Helmut Grabner, Michael Grabner, and Horst Bischof. Real-Time Tracking via On-line Boosting. In *British Machine Vision Conference*, pp. 1-6, 2006.

[37] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from RGB-D images for object detection and segmentation. In *European Conference on Computer Vision*, pp. 345-360, 2014.

[38] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pp. 297-312, 2014.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 37, pp. 1904–1916, 2015.

[40] Cher Keng Heng, Sumio Yokomitsu, Yuichi Matsumoto, and Hajime Tamura. Shrink boost for selecting multi-lbp histogram features in object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3250-3257, 2012.

[41] Van-Dung Hoang, My-Ha Le, and Kang-Hyun Jo. Hybrid cascade boosting machine using variant scale blocks based HOG features for pedestrian detection. In *Neurocomputing*, Vol. 135, pp. 357–366, 2014.

[42] Derek Hoiem, Alexei Efros, and Martial Hebert. Geometric context from a single image. In *IEEE International Conference on Computer Vision*, pp. 654-661, 2005.

[43] Xiaopeng Hong, Guoying Zhao, Haoyu Ren, and Xilin Chen. Efficient Boosted Weak Classifiers for Object Detection. In *Image Analysis*, pp. 205-214, 2013.

[44] Chang Huang, Haizhou Ai, Yuan Li, and Shihong Lao. Vector boosting for rotation invariant multi-view face detection. In *IEEE International Conference on Computer Vision*, pp. 446-453, 2005.

[45] Gary Huang, Marwan Mattar, Tamara Berg, and Erik Learned-Miller, . Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *Technical report*, University of Massachusetts, Amherst, 2007.

[46] Ahmad Humayun, Fuxin Li, and James Rehg. RIGOR: reusing inference in graph cuts for generating object regions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 336-343, 2014.

[47] Satoshi Ito and Susumu Kubota. Object classification using heterogeneous co-occurrence features. In *European Conference on Computer Vision*, pp. 701-714, 2010.

[48] Zdenek Kalal, Jiri Matas, and Krystian Mikolajczyk. Tracking-learning-detection. In *IEEE Conference on Pattern Analysis and Machine Intelligence*, Vol. 34, pp. 1409–1422, 2012.

[49] Leonid Karlinsky, Michael Dinerstein, Daniel Harari, and Shimon Ullman. The chains model for detecting parts by their context. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 25–32, 2010.

[50] Andrej Karpathy, Armand Joulin, and Fei-Fei Li. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in Neural Information Processing System*, pp. 1889-1897, 2014.

[51] Deok-Yeon Kim, Joon-Young Kwak, ByoungChul Ko, and Jae-Yeal Nam. Human detection using wavelet-based CS-LBP and a cascade of random forests. In *IEEE Conference on Multimedia Expo*, pp. 362–367, 2012.

[52] Merve Kilinc and Yusuf Akgul. Automatic human age estimation using overlapped age groups. In *Computer Vision, Imaging and Computer Graphics. Theory and Application*, pp. 313–325, 2013.

[53] Philipp Krähenbühl and Vladlen Koltun. Geodesic object proposals. In *European Conference on Computer Vision*, pp. 725-739, 2014.

[54] Christoph Lampert, Matthew Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2008.

[55] Ivan Laptev. Improving object detection with boosted histograms. In *International Journal on Computer Vision*, Vol. 27, pp. 535–544, 2009.

[56] Yann LeCun. What's wrong with deep learning? In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[57] Bastian Leibe, Aleš Leonardis, and Bernt Schiele. Robust object detection with interleaved categorization and segmentation. In *International Journal of Computer Vision*, Vol. 77, pp. 259-289, 2008.

[58] Dan Levi, Shai Silberstein, and Aharon Bar-Hillel. Fast multiple-part based object detection using kd-ferns. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 947-954, 2013.

[59] Tao Li, Mao Ye, and Jian Ding. Discriminative Hough context model for object detection. In *The Visual Computer*, Vol. 30, pp. 59–69, 2014.

[60] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *IEEE International Conference on Image Processing*, pp. 900-903, 2002.

[61] David Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision*, pp. 1150-1157, 1999.

[62] Tianyang Ma and Longin Latecki. From partial shape matching through local deformation to robust global shape similarity for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1441-1448, 2011.

[63] Subhransu Maji, Alexander Berg, and Jitendra Malik. Classification using intersection kernel support vector machines is efficient. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2008.

[64] Erno Makinen and Roope Raisamo. Evaluation of gender classification methods with automatically detected and aligned faces. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, pp. 541–547, 2008.

[65] Meredith Minear and Denise Park. A lifespan database of adult facial stimuli. In *Behavior Research Methods, Instruments, & Computers*, Vol. 36, pp. 630–633, 2004.

[66] Takeshi Mita, Toshimitsu Kaneko, and Osamu Hori. Joint haar-like features for face detection. In *IEEE International Conference on Computer Vision*, pp. 1619-1626, 2005.

[67] Jim Mutch and David Lowe. Multiclass object recognition with sparse, localized features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11-18, 2006.

[68] Woonhyun Nam, Piotr Dollár, and Joon Hee Han. Local Decorrelation for Improved Pedestrian Detection. In *Neural Information Processing Systems*, pp. 424-432, 2014.

[69] Duc Nguyen, Zhimin Zong, Philip Ogunbona, and Wanqing Li. Object detection using non-redundant local binary patterns. In *IEEE Conference on Image Processing*, pp. 4609–4612, 2010.

[70] Ryusuke Nosaka, Yasuhiro Ohkawa, and Kazuhiro, Fukui. Feature extraction based on co-occurrence of adjacent local binary patterns. In *Advances in Image and Video Technology*, pp. 82-91, 2012.

[71] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 24, pp. 971–987, 2002.

[72] Sakrapee Paisitkriangkrai, Chunhua Shen, and Anton Hengel. Strengthening the effectiveness of pedestrian detection with spatially pooled features. In *European Conference on Computer Vision*, pp. 546-561, 2014.

[73] Hong Pan, Yaping Zhu, and Liangzheng Xia. Efficient and accurate face detection using heterogeneous feature descriptors and feature selection. In *Computer Vision and Image Understanding*, Vol. 1, pp. 12–28, 2013.

[74] Ki-Yeong Park and Sun-Young Hwang. An improved Haar-like feature for efficient object detection. In *Pattern Recognition Letters*, Vol. 42, pp. 148–153, 2014.

[75] Jonathon Phillips, Hyeonjoon Moon, Syed Rizvi, and Patrick Rauss. The FERET evaluation methodology for face-recognition algorithms. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, pp. 1090–1104, 2000.

[76] Xianbiao Qi, Rong Xiao, Chun-Guang Li, Yu Qiao, Jun Guo, and Xiaoou Tang. Pairwise rotation invariant co-occurrence local binary pattern. In *European Conference on Computer Vision*, pp. 2199-2213, 2012.

[77] Shengsheng Qian, Tianzhu Zhang, and Changsheng Xu. Boosted multi-modal supervised latent dirichlet allocation for social event classification. In *IEEE International Conference on Pattern Recognition*, pp. 1999-2004, 2014.

[78] Nikhil Rasiwasia and Nuno Vasconcelos. Holistic context modeling using semantic co-occurrences. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1889-1895, 2009.

[79] Haoyu Ren, Cher-Keng Heng, Wei Zheng, Luhong Liang, and Xilin Chen. Fast object detection using boosted co-occurrence histograms of oriented gradients. In *IEEE International Conference on Image Processing*, pp. 2705-2708, 2010.

[80] Haoyu Ren and Ze-Nian Li. Basis Mapping Based Boosting for Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1583-1591, 2015.

[81] Haoyu Ren and Ze-Nian Li. Boosted local binaries for object detection. In *IEEE Internation Conference on Multimedia and Expo*, pp. 1-6, 2014.

[82] Haoyu Ren and Ze-Nian Li. Object detection using edge histogram of oriented gradient. In *IEEE International Conference on Image Processing*, pp. 4057-4061, 2014.

[83] Haoyu Ren and Ze-Nian Li. Object Tracking Using Structure-aware Binary Features. In *IEEE Internation Conference on Multimedia and Expo*, pp. 1-6, 2015.

[84] Haoyu Ren and Ze-Nian Li. Object Recognition Based on Deformable Contour Set. In *IEEE International Conference on Image Processing*, pp. 1427-1431, 2015.

[85] Haoyu Ren and Ze-Nian Li. Object detection using generalization and efficiency balance co-occurrence features. In *IEEE International Conference on Computer Vision*, 1944-1952, 2015.

[86] Haoyu Ren, Xiaopeng Hong, Cher-Keng Heng, Luhong Liang, and Xilin Chen. A Sample Pre-mapping Method Enhancing Boosting for Object Detection. In *International Conference on Pattern Recognition*, pp. 3005-3008, 2010.

[87] Henry Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 20, pp. 23–38, 1998.

[88] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: an efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision*, pp. 2564-2571, 2011.

[89] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, and Michael Bernstein. Imagenet large scale visual recognition challenge. In *International Journal of Computer Vision*, Vol. 115, pp. 211-252, 2015.

[90] Bryan Russell, William Freeman, Alexei Efros, Josef Sivic, and Andrew, Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1605-1614, 2006.

[91] Mohammad Saberian and Nuno Vasconcelos. Learning optimal embedded cascades. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 34, pp. 2005–2018, 2012.

[92] Jakob Santner, Christian Leistner, Amir Saffari, Thomas Pock, and Horst Bischof. Prost: Parallel robust online simple tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 723-730, 2010.

[93] Robert Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. In *Machine learning*, Vol. 37, pp. 297–336, 1999.

[94] Robert Schapire, Yoav Freund, Peter Bartlett, and Wee-Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Annals of Statistics*, pp. 1651-1686, 1998.

[95] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *arXiv:1312.6229*, 2013.

[96] Caifeng Shan. Learning local binary patterns for gender classification on real-world face images. In *Pattern Recognition Letters*, Vol. 33, pp. 431–437, 2012.

[97] Jamie Shotton, Andrew Blake, and Roberto Cipolla. Multiscale categorical object recognition using contour fragments. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, pp. 1270–1281, 2008.

[98] Hyun Song, Ross Girshick, Stefanie Jegelka, Julien Mairal, Zaid Harchaoui, and Trevor Darrell. On learning to localize objects with minimal supervision. In *arXiv:1403.1024*, 2014.

[99] Luciano Spinello and Kai Arras. Leveraging RGB-D data: Adaptive fusion and domain adaptation for object detection. In *IEEE Conference on Robotics and Automation*, pp. 4469-4474, 2012.

[100] Praveen Srinivasan, Qihui Zhu, and Jianbo Shi. Many-to-one contour matching for describing and discriminating object shape. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1673-1680, 2010.

[101] Nitish Srivastava and Ruslan Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Advances in Neural Information Processing Systems*, pp. 2222-2230, 2012.

[102] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In *Advanced Neural Information Processing System*, pp. 2553-2561, 2013.

[103] Christian Szegedy, Scott Reed, Dumitru Erhan, and Dragomir Anguelov. Scalable, high-quality object detection. In *arXiv:1412.1441*, 2014.

[104] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-9, 2015.

[105] Yichuan Tang, Ruslan Salakhutdinov, and Geoffrey Hinton. Robust boltzmann machines for recognition and denoising. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2264-2271, 2012.

[106] Juan Tapia, Flores Pérez , and Claudio André. Gender classification based on fusion of different spatial scale features selected by mutual information from histogram of LBP, intensity, and shape. In *IEEE transactions on information forensics and security*, Vol. 8, pp. 488–499, 2013.

[107] Zhuowen Tu. Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering. In *IEEE International Conference on Computer Vision*, pp. 1589-1596, 2005.

[108] Oncel Tuzel, Fatih Porikli, and Peter Meer. Region covariance: A fast descriptor for detection and classification. In *European Conference on Computer Vision*, pp. 589-600, 2006.

[109] Oncel Tuzel, Fatih Porikli, and Peter Meer. Pedestrian detection via classification on riemannian manifolds. In *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 30, pp. 1713–1727, 2008.

[110] Jasper Uijlings, Koen Sande, Theo Gevers, and Arnold Smeulders. Selective search for object recognition. In *International journal of computer vision*, Vol. 104, pp. 154-171, 2013.

[111] Régis Vaillant, Christophe Monrocq, and Yann LeCun. Original approach for the localisation of objects in images. In *IEEE Proceedings of Vision, Image and Signal Processing*, pp. 245-250, 1994.

[112] Vladimir Vapnik. The nature of statistical learning theory. 2013.

[113] Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, pp.480–492, 2012.

[114] Paul Viola and Michael Jones. Robust real-time face detection. In *International Journal on Computer Vision*, Vol. 57, pp. 137–154, 2004.

[115] Carl Vondrick, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. Hoggles: Visualizing object detection features. In *IEEE International Conference on Computer Vision*, pp. 1–8, 2013.

[116] Xinggang Wang, Xiang Bai, Tianyang Ma, Wenyu Liu, and Longin Latecki. Fan shape model for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 151-158, 2012.

[117] Xiaoyu Wang, Tony Han, and Shuicheng Yan. An HOG-LBP human detector with partial occlusion handling. In *IEEE International Conference on Computer Vision*, pp. 32-39, 2009.

[118] Xiaoyu Wang, Ming Yang, Shenghuo Zhu, and Yuanqing Lin. Regionlets for generic object detection. In *IEEE International Conference on Computer Vision*, pp. 17-24, 2011.

[119] Tomoki Watanabe, Satoshi Ito, and Kentaro Yokoi. Co-occurrence histograms of oriented gradients for pedestrian detection. In *Advances in Image and Video Technology*, pp. 37-47, 2009.

[120] Christian Wojek and Bernt Schiele. A performance evaluation of single and multi-feature people detection. In *Pattern Recognition*, Vol. 32, pp. 82–91, 2008.

[121] Bo Wu and Ram Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. In *International Journal on Computer Vision*, Vol. 75, pp. 247–266, 2007.

[122] Bo Wu and Ram Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. In *IEEE International Conference on Computer Vision*, pp. 1-8, 2007.

[123] Bo Wu, and Ram Nevatia. Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection? In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[124] Jianxin Wu. A fast dual method for HIK SVM learning. In *European Conference on Computer Vision*, pp. 552-565, 2010.

[125] Jianxin Wu. Efficient HIK SVM learning for image classification. In *IEEE Transaction on Image Processing*, Vol. 21, pp. 4442–4453, 2012.

[126] Jianxin Wu, Charles Brubaker, Matthew Mullin, and James Rehg. Fast asymmetric learning for cascade face detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, pp. 369–382, 2013.

[127] Jingsong Xu, Qiang Wu, Jian Zhang, and Zhenmin Tang. Object detection based on co-occurrence gmulbp features. In *IEEE Internation Conference on Multimedia and Expo*, pp. 943-948, 2012.

[128] Jingsong Xu, Qiang Wu, Jian Zhang, Fumin Shen, and Zhenmin Tang. Boosting Separability in Semi-supervised Learning for Object Classification. In *IEEE International Conference on Pattern Recognition*, pp. 1197-1208, 2014.

[129] Junjie Yan, Xucong Zhang, Zhen Lei, Shengcai Liao, and Stan Li. Robust multi-resolution pedestrian detection in traffic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3033-3040, 2013.

[130] Shengye Yan, Shiguang Shan, Xilin Chen, and Wen Gao. Locally assembled binary (lab) feature with feature-centric cascade for fast and accurate face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-7, 2008.

[131] Shulin Yang, Mei Chen, Dean Pomerleau, and Rahul Sukthankar. Food recognition using statistics of pairwise local features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2249-2256, 2010.

[132] Yi Yang and Shawn Newsam. Spatial pyramid co-occurrence for image classification. In *European Conference on Computer Vision*, pp. 1465-1472, 2011.

[133] Junsong Yuan, Ming Yang, and Ying Wu. Mining discriminative co-occurrence patterns for visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2777-2784, 2011.

[134] Yimeng Zhang, Zhaoyin Jia, and Tsuhan Chen. Image retrieval with geometry-preserving visual phrases. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 809-816, 2011.

[135] Junge Zhang, Kaiqi Huang, Yinan Yu, and Tieniu Tan. Boosted local structured hog-lbp for object localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1393-1400, 2011.

[136] Kaihua Zhang, Lei Zhang, and Ming-Hsuan Yang. Real-time compressive tracking. In *European Conference on Computer Vision*, pp. 864-877, 2012.

[137] Wei Zheng, Hong Chang, Luhong Liang, Haoyu Ren, Shiguang Shan, and Xilin Chen. Strip features for fast object detection. In *International Journal on Computer Vision*, Vol. 42, pp. 1898–1912, 2013.

[138] Wei Zheng and Luhong Liang. Fast car detection using image strip features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2703-2710, 2009.

[139] Long Zhu, Yuanhao Chen, Alan Yuille, and William Freeman. Latent hierarchical structural learning for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1062-1069, 2010.

[140] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1491-1498, 2006.

[141] Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pp. 391-405, 2014.